NO. 52 SEPTEMBER 1982

U.S./Canada Edition: $2.50
International Edition: $2.95
United Kingdom Edition: £1.80

MICRO™

THE 6502/6809 JOURNAL

68000 Feature

A/D Conversion on the Atari

Superimposing TV Pictures with the PET

Apple Program Compressor

WHY THE MICROSOFT RAMCARD™ MAKES OUR SOFTCARD™ AN EVEN BETTER IDEA.

Memory — you never seem to have quite enough of it.

But if you're one of the thousands of Apple owners using the SoftCard, there's an economical new way to expand your memory dramatically.

16K ON A PLUG-IN CARD.

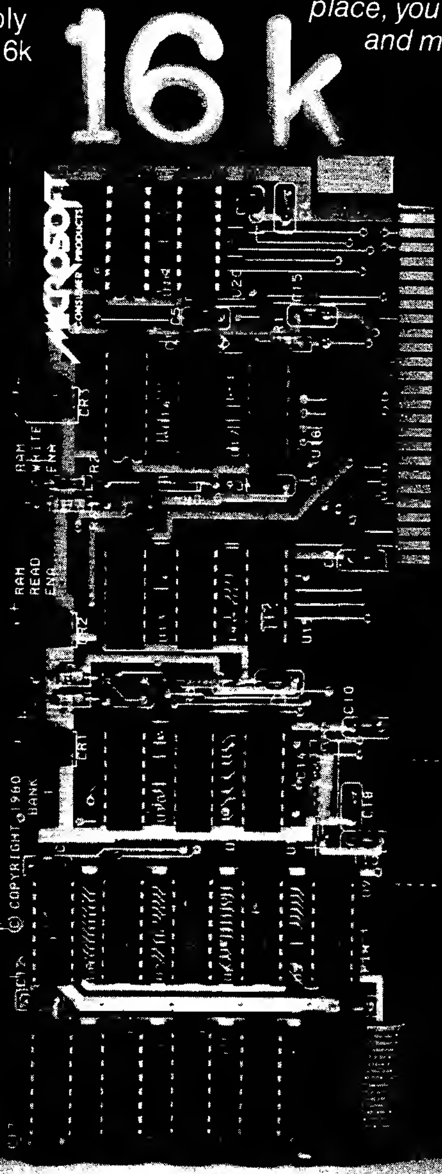
Microsoft's new RAMCard simply plugs into your Apple II,® and adds 16k bytes of dependable, buffered read/write storage.

Together with the SoftCard, the RAMCard gives you a 56k CP/M® system that's big enough to take on all kinds of chores that would never fit before (until now, the only way to get this much memory was to have an Apple Language Card installed).

GREAT SOFTWARE: YOURS, OURS, OR THEIRS.

With the RAMCard and SoftCard, you can tackle large-scale business and scientific computing with our COBOL and FORTRAN languages. Or greatly increase the capability of CP/M applications like the Peachtree Software accounting systems. VisiCalc™ and other Apple software packages can take advantage of RAMCard too.

And RAMCard gives you the extra capacity to develop advanced programs of your own, using the SoftCard and CP/M. *Even with the RAMCard in place, you can still access your ROM BASIC and monitor routines.*



JOIN THE SOFTCARD FAMILY.

The RAMCard is just the latest addition to the SoftCard family — a comprehensive system of hardware and software that can make your Apple more versatile and powerful than you ever imagined.

Your Microsoft dealer has all the exciting details. Visit him soon, and discover a great idea that keeps getting better.

Microsoft Consumer Products
10700 Northup Way
Bellevue, WA 98004
206-828-8080

MICROSOFT™

2MHZ 6809 SYSTEMS

GIMIX offers you a variety to choose from!

38 MB WINCHESTER SYSTEM $17,498.99

HARDWARE FEATURES:

- ★ 2MHz 6809 CPU
- ★ 512KB Static RAM
- ★ 8 RS232C Serial Ports
- ★ 2 Parallel Ports
- ★ DMA Double Density Floppy Disk Controller
- ★ Dual 8" DSDD Floppy Disk System
- ★ Dual Winchester Subsystem with Two19 MB 5¼" Winchester Drives

SOFTWARE FEATURES:

- ★ OS-9 LEVEL TWO Multi-User Operating System
- ★ OS-9 Debugger
- ★ OS-9 Text Editor
- ★ OS-9 Assembler

19 MB WINCHESTER SYSTEM $8998.09

HARDWARE FEATURES:

- ★ 128K Static Ram
- ★ 2MHz 6809 CPU
- ★ 19 MB 5¼" Winchester DMA Subsystem
- ★ 4 RS232C Serial Ports
- ★ 1 MB 5¼" Floppy Disk Drive
- ★ DMA Double Density Floppy Disk Controller

SOFTWARE FEATURES:

- ★ OS-9 LEVEL TWO Multi-User Operating System
- ★ OS-9 Text Editor
- ★ OS-9 Debugger
- ★ OS-9 Assembler

128KB MULTI-USER SYSTEM $6997.39

HARDWARE FEATURES:

- ★ 2MHz 6809 CPU
- ★ DMA Double Density Floppy Disk Controller
- ★ 128KB Static Ram
- ★ 2 RS232C Serial Ports
- ★ Dual 8" DSDD Floppy Disk System

SOFTWARE FEATURES: Your choice of either UniFLEX or OS-9 LEVEL TWO. Both are Unix-like Multi-User/Multi-Tasking Operating Systems.

56KB FLEX/OS-9 "SWITCHING" SYSTEM $4148.49

HARDWARE FEATURES:

- ★ 2MHz 6809 CPU
- ★ 56K Static Ram
- ★ 2 RS232C Serial Ports
- ★ DMA Double Density Floppy Disk Controller
- ★ 2 Built-in 5¼" 40tr DSDD Disk Drives (80 Track DSDD Drive Option . . add $400.00)

SOFTWARE FEATURES:

- ★ GMXBUG monitor — FLEX Disk Operating System
- ★ OS-9 LEVEL ONE Multi-tasking operating system for up to 56K of memory

WINCHESTER SUBSYSTEMS

Winchester packages are available for upgrading current **GIMIX** 6809 systems equipped with DMA controllers, at least one floppy disk drive, and running FLEX, OS-9 LEVEL ONE or OS-9 LEVEL TWO. The packages include one or two 19MB (unformatted) Winchester drives, DMA Hard Disk Interface, and the appropriate software drivers. The Interface can handle two 5¼" Winchester Drives, providing Automatic Data Error Detection and Correction: up to 22 bit burst error detection and 11 bit burst error correction.

Dual drives can be used together to provide over 30 MBytes of on line storage -- or use one for back-up of the other. (More convenient and reliable than tape backup systems.

#90 includes one 19MB Drive, Interface, and Software	**$4288.90**
#91 includes two 19MB Drives, Interface, and Software.	**$6688.91**

Contact GIMIX for systems customized to your needs or for more information.

50 HZ Export Versions Available

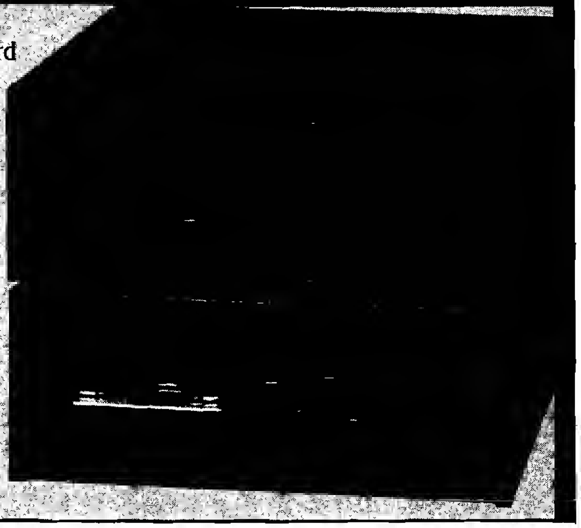
GIMIX Inc. reserves the right to change pricing and product specifications at any time without further notice.

GIMIX® and GHOST® are registered trademarks of GIMIX Inc
FLEX and UniFLEX are trademarks of Technical Systems Consultants Inc.
OS-9 is a trademark of Microware Inc.

1337 WEST 37th PLACE
CHICAGO, ILLINOIS 60609
(312) 927-5510
TWX 910-221-4055

GIMIX inc.

1982 GIMIX Inc.



TM

September Highlights

68000 Feature

Once again the microcomputer industry has a new and exciting product to offer users: the 68000 microprocessor. Dozens of companies have already built computers around this chip, or developed compatible software and peripherals. The big news about the 68000 is that it can be used as a powerful add-on to your 6502.

Laurence Kepple, in "The 68000 and the Personal Computer" (p. 27), discusses the 68000 as a 6502 add-on. He offers some insights on the manufacturers and their products. If you're interested in adding a 68000 to your present system, take a look at this article.

We're providing a very useful and detailed series on the 68000 microprocessor itself, authored by Joe Hootman, a Professor of Electrical Engineering at the University of North Dakota. The first part (p. 41) contains a brief introduction to the 68000 and detailed treatment of the data movement commands. Future installments will offer tables on the 68000's instruction set.

Tom Whiteside and Joe Jelemensky of Motorola, manufacturer of the 68000, describe the registers, instruction set, and addressing modes of this microprocessor (p. 32).

Our data sheet this month will serve as a handy reference to 68000 programmers.

6809

Hal Clark presents an article (p. 57) for 6809-based machines to give a Pascal-like structure to your machine-language programs. This makes the program easier to read and debug while maintaining the execution speed and memory efficiency of assembly language.

A BASIC program to calculate market projections on industry and company sales, by Len Suckle, is presented for the Color Computer (p. 67). It could be translated to other BASICs fairly easily and is well documented so you could customize for a specific application.

Apple

For Apple users, Barton Bauers has written an extremely useful machine-language utility to reduce the length of Applesoft BASIC programs and increase the speed of loading and execution (p. 89). It allows you to fully document your program with remarks and single line statements for legibility, and then remove the remarks and concatenate all non-referenced lines for a streamlined version.

This month's "Apple Slices" column (p. 47) by Tim Osborn, presents a quick random-access technique called "hashing." If you want to speed up access to your file programs that do a lot of random accessing, here is the solution.

We welcome Dr. Richard C. Vile, Jr., as an Apple contributing editor. Richard will be working with our in-house Apple specialist, Phil Daley, to expand our Apple coverage.

Commodore

There are two articles of special interest to PET/CBM owners. Peter Hiscocks (p. 11) describes a hardware technique to simultaneously display on the PET's screen the computer's output and the image from a closed-circuit TV camera. Although the PET does not have access to the image, it can be programmed to display fancy captions or to point out certain features. The picture could also be used to enhance an otherwise dull computer run.

"Auto SAVE for the PET" by Louis Sander (p. 83) is a BASIC subroutine you attach to your programs during development. It ensures that you make copies of your program at regular intervals, thereby giving you more protection from unanticipated disasters.

Loren Wright's "PET Vet" column (p. 78) discusses Eastern House Software's MAE macro assembler and some new Commodore-oriented publications.

We would like to welcome Dave Malmberg and Jim Strasma as contributing editors in the Commodore area.

Atari

We are happy to have Paul Swanson on board now as a contributing editor and Atari columnist. Paul's column will begin in November. He will also help us increase our base of Atari authors.

Straightforward Garbage Collection

This program by Cornelis Bongers appeared in our August issue, page 90 (also one of the utilities on disk). A few readers missed the important note that appeared at the end of Mr. Bongers' article. Here it is again, in case you missed it too:

The machine-language program starts at $9000 and has a length of $260 bytes. After assembling the text file and storing it to disk, the program can be installed by : BRUN programname. This command executes an initialization routine that sets HIMEM to $9000 and installs the & vector. If you want to BRUN the routine from within an Applesoft program, the BRUN command should be inserted at the first line of the program, and must be followed by a CLEAR command. For example:

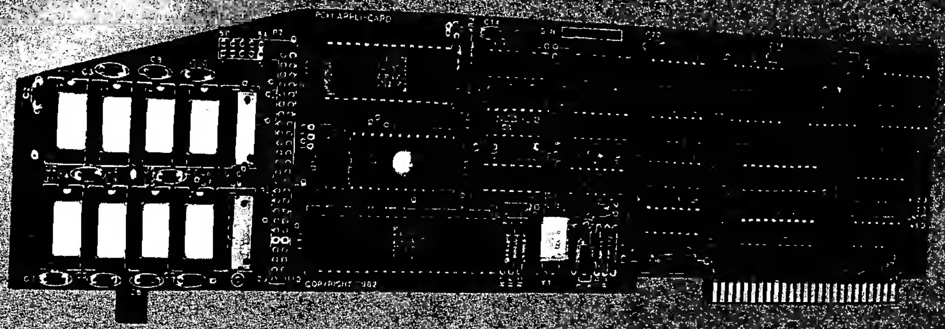
```
10 PRINT "BRUN program name": CLEAR: REM control D
   behind first quotes.
```

The program makes use of some Applesoft routines in ROM. If the RAM version of Applesoft is being used, the relevant subroutine calls have to be adjusted. MICRO™

NEW . . . For Apple II & III*
from **PERSONAL COMPUTER PRODUCTS, INC.**

APPLI-CARD

4 or 6 mhz Z-80 and 64K ON-CARD MEMORY



THE ONE CARD SOLUTION TO EXECUTE CP/M* APPLICATION [illegible]

- Development Languages Available
- Applications Available
- Compare Our Features And Get The Best Value And Performance For Your Money

FEATURES	Z-CARD*	SoftCard*	APPLI-CARD
6 mhz Z-80 available	No	No	Yes
64K on-card memory	No	No	Yes
CP/M available	Yes	Yes	Yes
SB-80* with card	No	No	Yes
40 col. to 255 col. horizontal scroll	No	No	Yes
Choice of application	No	No	Yes
2K PROM on the card	No	No	Yes
Real time clock available on the card	No	No	Yes
Expansion interface on the card	No	No	Yes
70 col. upper & lower case	No	No	Yes
A self-contained Z-80A or Z-80B with memory	No	No	Yes
One-card Wordstar* execution	No	No	Yes
63K available for program development or execution	No	No	Yes
Menu driven set up	No	No	Yes

Call today for pricing and product information. Dealer inquiries invited.

PERSONAL COMPUTER [illegible]
[illegible] Bernardo Center [illegible]
San Diego, California [illegible]
(619) [illegible] or Call Your [illegible]

*Registered Trade Marks: Apple II & III Apple Computer [illegible] ARD for Apple III will be [illegible]
Z-CARD Advanced Logic Systems; SoftCard Microsoft [illegible] Products; SB-80 [illegible] Associates [illegible]

NO. 52 SEPTEMBER 1982

MICRO™

THE 6502/6809 JOURNAL

STAFF

President/Editor-in-Chief
ROBERT M. TRIPP

Publisher
MARY GRACE SMITH

Senior Editor
LAURENCE KEPPLE

Editorial Staff
PHIL DALEY — Technical editor
JOHN HEDDERMAN — Jr. programmer
MARJORIE MORSE — Editor
JOAN WITHAM — Editorial assistant
LOREN WRIGHT — Technical editor

Graphics Department
HELEN BETZ — Director
PAULA M. KRAMER — Production mgr.
EMMALYN H. BENTLEY — Typesetter

Sales and Marketing
CATHI BLAND — Advertising mgr.
CAROL A. STARK — Circulation mgr.
LINDA HENSDILL — Dealer Sales
MAUREEN DUBE — Promotion

Accounting Department
DONNA M. TRIPP — Comptroller
KAY COLLINS — Bookkeeper
EILEEN ENOS — Bookkeeper

Contributing Editors
DAVE MALMBERG
JIM STRASMA
PAUL SWANSON
RICHARD VILE

Advertising Sales Representatives
See Page 127

Subscription/Dealer Inquiries
(617) 256-5515

DEPARTMENTS

68000 FEATURE

HARDWARE

LANGUAGE AIDS

APPLICATIONS

UTILITIES

COLUMNS

SOUTHWESTERN DATA SYSTEMS

PROUDLY INTRODUCES ONE OF THE **TASTIEST** MACHINE LANGUAGE DEVELOPMENT SYSTEMS AVAILABLE . . .

"THE S•D•S COMBO!"

[illegible]

[illegible] The Book

by [illegible]

A BEGINNER'S GUIDE TO 6502 PROGRAMMING ON THE APPLE II.

For beginners, ASSEMBLY LINES: THE BOOK provides a clear and non-technical introduction to machine language programming on the Apple. Drawn from the monthly column in Softalk Magazine, and expanded to provide even more information, ASSEMBLY LINES: THE BOOK has already received critical acclaim as the best tutorial on machine language programming available.

Example programs include paddles, sound, disk files and more, all presented with the novice programmer in mind. The book also includes an excellent reference section listing each machine language command, and a sample listing illustrating its most common uses in actual source listing.

SUGGESTED RETAIL PRICE: $19.95

Merlin By Glen Bredon

A 6502 MACRO ASSEMBLER

For programmers of all skill levels, we think you'll find that MERLIN is the most powerful 6502 assembler available for the Apple, while at the same time being the easiest to use.

IN FACT, WE'RE SO CONFIDENT OF THIS CLAIM, THAT IF YOU CAN FIND A BETTER ASSEMBLER WITHIN 30 DAYS OF PURCHASE, SIMPLY RETURN THE COMPLETE MERLIN PACKAGE FOR A FULL REFUND!

A full featured macro assembler with optional assembly to disk, and use of 'include' files. Merlin's editor has word processor-like power with such options as global search/replace, a powerful line editor, and more. The package also includes SOURCEROR, a utility to generate labeled pseudo source code from raw binary files, and **also SOURCEROR.FP, a fully labeled and commented source listing of Applesoft BASIC!**

SUGGESTED RETAIL PRICE: $64.95

munch-a-bug

By [illegible]

A 6502 [illegible]

To [illegible] MUNCH-A-BUG. The [illegible] means to walk through [illegible] programs one step at a time. [illegible] educational to the beginner, and an invaluable de-bugging aid to the advanced programmer. In addition, the same money-back guarantee applies to MUNCH-A-BUG as to MERLIN!

More than a simple step and trace utility, MUNCH-A-BUG includes its own mini-assembler, supports labels, and even conditional trace flags. This means M.A.B. can be put in a 'dormant' state which will later 'pop-up' in the trace mode, only when certain conditions are met. Thus routines within fully operational programs can be tested right at run-time.

SUGGESTED RETAIL PRICE: $49.95

SPECIAL LIMITED TIME COMBINATION PRICE: $119.95

THIS OFFER VALID THROUGH OCTOBER 31, 1982. • CALIFORNIA RESIDENTS ADD 6% SALES TAX

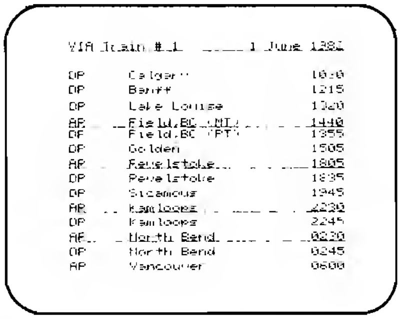
Please ask your local Apple dealer for more details, or write SDS for a sample list of MERLIN's commands and a complete product guide of over 20 other outstanding programs!

*MERLIN is the assembler of choice of these leading software companies: ARTSCI, INC. • BRODERBUND SOFTWARE, INC. • GEBELLI SOFTWARE, INC. • PROMETHEUS PRODUCTS, INC. • SIRIUS SOFTWARE, INC. • SYNERGISTIC SOFTWARE.

southwestern data systems™

P.O. BOX 582 • SANTEE, CALIFORNIA 92071 • TELEPHONE: 714/562-3670

About the Cover



```
VIA Train #1             1 June 1982

DP      Calgary                 1030
DP      Banff                   1215
DP      Lake Louise             1320
AR      Field,BC (MT)           1440
DP      Field,BC (PT)           1355
DP      Golden                  1505
AR      Revelstoke              1805
DP      Revelstoke              1835
DP      Sicamous                1945
AR      Kamloops                2230
DP      Kamloops                2245
AR      North Bend              0220
DP      North Bend              0245
AR      Vancouver               0600
```

VIA Train Number 1, "The Canadian," makes its way westward across the continental divide. Nearly three days ago it left Montreal and tomorrow morning it will be in Vancouver, British Columbia.

Computers are used by travel agencies to keep transportation schedules, such as this one, readily accessible. A microcomputer could be installed on board the train and tied into a radio communications network with other, larger computers. The Canadian Pacific Railroad (whose tracks VIA is using) runs long sulfur, coal, and grain trains with computer-controlled "robot" locomotives midway through the train. Every change in speed or gear made by the crew at the head of the train is duplicated by the robot locomotives.

Photo by Loren Wright, MICRO Editor

MICRO is published monthly by:
MICRO INK, Chelmsford, MA 01824
Second Class postage paid at:
Chelmsford, MA 01824 and additional mailing offices
USPS Publication Number: 483470
ISSN: 0271-9002

Send subscriptions, change of address, USPS Form 3579, requests for back issues and all other fulfillment questions to

MICRO INK
34 Chelmsford Street
P.O. Box 6502
Chelmsford, MA 01824

or call
617/256-5515
Telex: 955329 TLX SRVC
800-227-1617

Subscription Rates	*Per Year*
U.S.	$24.00 2 yr. / $42.00
Foreign surface mail	$27.00
Air mail:	
Europe	$42.00
Mexico, Central America, Middle East, North Africa, Central Africa	$48.00
South America, South Africa, Far East, Australasia, New Zealand	$72.00

Editorial

When Worlds Collide

The 68000 microprocessor is one of a number of new chips that are making it increasingly difficult for minicomputer manufacturers to justify their existence as an enterprise distinct from the world of microcomputing. More than market share and terminology are at stake. The critical issue is: what role does the individual user play as part of a computer system? In a microcomputer system, the computer and the user are on equal terms, one-on-one partners in a problem-solving endeavor. In a minicomputer, multiuser environment, the individual user is simply less important. Many users must compete for the attention of the CPU. Waiting in line has never been popular, and some people are asking why, in this era of ever cheaper hardware, there are those who say we should go on waiting. For example Hal Hardenbergh, in his newsletter *DTACK Grounded*, speaks up for the individual user with some vigor: "Refusing to give each user his or her own individual, undivided processor chip is prima-facie evidence of utter contempt for the customer. And utter contempt is all that a minicomputer type has for us personal computer types!"

"Utter contempt"? Surely Mr. Hardenbergh goes too far. However, consider the following observation from Ms. Andrea Lewis in a recent issue of *Computer Design*: "Putting a minicomputer operating system on a toy computer will not result in a cheap minicomputer." By "toy computer," Ms. Lewis means to indicate "microcomputer." She goes on, "To support multiple users in a true timesharing environment, reliable memory protection must be provided to keep one errant user from disrupting the other users' work on the system." A "true timesharing environment"? Is there something intrinsically valuable about timesharing? Useful when hardware was fabulously expensive, timesharing still requires costly overhead, such as the "reliable memory protection" to which Ms. Lewis refers.

Other costs of timesharing are more subtle. For example, VisiCalc, a program that was conceived especially for "toy computers," does not work well in a timesharing environment. As Tracy Licklider, Vice President of Software Arts, noted recently, "VisiCalc is a screen-intensive, highly interactive program. Timesharing systems simply cannot process VisiCalc's frequent demands for screen refreshes. In a minicomputer, multiuser environment, VisiCalc gets bogged down." Thus, one of the most powerful programs in history succeeded in part because its designers were thinking in terms of a one-on-one relationship between the computer and its user.

Interactivity is the key, because interactivity means maximum user involvement. What minicomputer manufacturers have yet to understand is that the more involvement users have in a computer system, the more powerful the computer system becomes. As Hal Hardenbergh puts it: "To a minicomputer type, a minicomputer which has seventeen applications programs available for it is described as 'heavily supported.' You tell the owner of a PET, Apple, or [TRS] 80 that a particular personal computer has a *thousand* application programs available and they will reply, 'Oh? What's wrong with it? Isn't it a popular machine?' " The fantastic support available for a micro like the Apple is a direct result of user involvement.

Even those with a minicomputer orientation are beginning to think in terms of "one person, one processor." *Computer Business News* recently quoted Motorola Inc.'s Tom Starnes as saying that, though everyone does not need distributed processing capabilities in their applications now, almost everyone will "run out of gas" in terms of processing capability using a single system bus. "Which is why applications are turning more frequently to distributed processing where a number of different processors each performs their own isolated programs," he contended. As for timesharing systems, Starnes said, "It is not much more expensive to have a [16-bit chip] and memory for user programs in the workstation itself. The benefit is the user has a high-performance processor all to himself." To which we at MICRO say, "Amen."

Laurence Kepple

Letters/Updates

Atari DOS Bug

Dear Editor:

There is a bug in the Atari DOS (or BASIC's disk I/O) that can clobber up to half of RAM page 6, the "safe" ram. Since the operation of this bug isn't obvious, it can cause incredible amounts of trouble; I spent two days going over a USR subroutine to try to find the bug, before I discovered that my routine had nothing to do with it.

A bug in either DOS (DOS I, DOS 2.0's, and OS/A tested), the BASIC cartridge, or the 10K OS ROM (Version A tested), causes the location of the disks' I/O buffer to move from around location 7000-8000 (decimal) to location 1408, which is only 128 bytes below location 1536, the beginning of page six.

This means that an input or output of more than 128 bytes will destroy some or all of the first 127 bytes of page six!

David H. Simmons
P.O. Box 7000-140
Redondo Beach, CA 90277

Atari Bug Demo

```
TESTING INSTRUCTIONS:
A: Type in program.
B: SAVE to disk as "D:BUGTEST.BAS".
C: Type "RUN".

0100 REM :        BUG-DEMO
0110 REM : BY DAVID H. SIMMONS
0120 DIM IN$(256)
0130 OPEN #1,4,0,"D:BUGTEST.BAS"
0140 REM (OR OTHER 'TEXT RECORD' FILE)
0150 ? "IOCB BUFFER ADDRESS: ";PEEK(852)+256*PEEK(853)
0160 REM : *(IOCB #1 ICBAL/ICBAH)*
0170 INPUT #1;IN$:INPUT #1;IN$
0180 ? "AFTER INPUT: ";PEEK(852)+256*PEEK(853)
0190 ? " +128= 1536, START OF PAGE SIX!":? "I/O Of more than
 128 bytes WIPES OUT anything in page six!"
0200 CLOSE #1:END
```

Othello Update

Phil Daley, MICRO's Technical Editor for the Apple, offers this update:

In the program Othello (*MICRO on the Apple*, Vol. 3), Charles Taylor presents a two-person game with the computer keeping track of the board position and legality of moves. With the minor modifications presented here, the computer will play a level 1 game of Othello for those who need an opponent. This would be especially useful in learning how to play the game.

Lines 16-135 contain adjustments to the Taylor program corresponding lines. Lines 173-183 are new material. There was a minor bug in the program as published. The only way the program knew that the game ended was when all 64 blocks were filled. It is possible to not have any moves left when there are still unfilled spaces. Line 135 allows a player to enter "DONE" when this condition occurs and thereby end the game.

If you wish to design a level 2 game, a subroutine to examine how many "captures" could occur on each possible move would improve the computer's chances.

Listing for Othello Line Changes

```
16   DIM BOARD(9,9): DIM COMPUTERMOVE$(61)
34   FOR I = 1 TO 8: READ DY(I): NEXT I: FOR I = 1 TO 61: READ CO$(I): NEXT

56 TURN = 2: GOSUB 181
60  COLOR= CC(TURN):ZZ = 0
62  IF  NOT Z OR TURN = 1 THEN  PRINT PROMPT$(TURN)
133  IF Z AND TURN = 2 THEN ZZ = ZZ + 1:MOVE$ = CO$(ZZ): GOTO 135: REM
                SUBR GETMOVE
135 PASS = 0: IF MOVE$ = "DONE" THEN PASS = 1:Q = 100: RETURN
173  DATA A1,A8,H1,H8,B1,B8,G1,G8
174  DATA A2,A7,H2,H7,C1,C8,F1,F8
175  DATA A3,A6,H3,H6,D1,D8,E1,E8
176  DATA A4,A5,H4,H5,C4,C5,F4,F5
177  DATA D3,D6,E3,E6,C3,C6,F3,F6
178  DATA D2,D7,E2,E7,B4,B5,G4,G5
179  DATA B3,B6,C2,C7,F2,F7,G3,G6
180  DATA B2,B7,G2,G7,P
181  HOME : PRINT "DO YOU WISH TO PLAY THE COMPUTER?"
182  INPUT Z$: IF  LEFT$ (Z$,1) = "Y" THEN Z = 1
183  RETURN
```

Fan Improvement

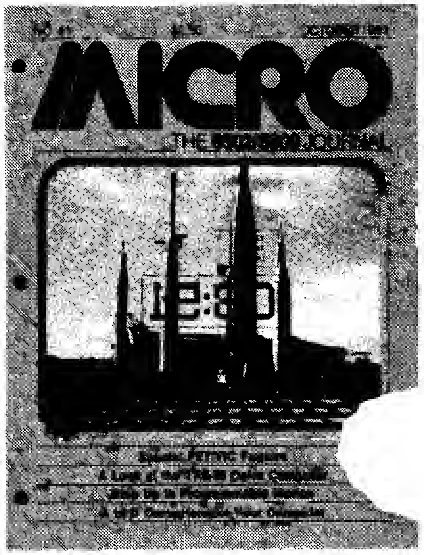
Dear Editor:

I recently purchased a System Saver fan by Kensington Software. Although the fan works as advertised, I have made two improvements that your readers might be interested in.

First I covered the vent slots on the right side of my Apple with a section of air-conditioning filter. This prevents dust from being drawn across the inside of the computer. The filter was attached with duct tape to the outside of the computer.

The second improvement was to place 1/8-inch thick foam rubber between the part of the fan case and computer that come in contact with each other. This minimizes vibration.

Robert Gershowitz
76-51 169 Street
Flushing, NY 11366



Take a Bigger Byte of MICRO

Special Introductory Offer

Subscribe Now and Save 33%
(Off Newsstand Price)

(First Time Subscribers Only)

MICRO brings you information-packed pages every month.
Each issue delivered to your door opens doors to computing mastery!

Subscribe to MICRO and learn how to extend your computing knowledge to get the most out of your computer and expand its capabilities.

Subscribe to MICRO and develop your microcomputing skill. Keep up with the latest technology—Don't be left behind!

Subscribe to MICRO Today and find, each month at your door, a world of computing power you could only dream of yesterday.

All for Only

$19.95

1 YEAR
SUBSCRIPTION

MICRO
for the
SERIOUS
COMPUTERIST

83-356

□ YES! I'm Serious
Send me a 1 Year Introductory Subscription
for only **$19.95** (Foreign—surface mail—**$22.95**)

MICRO
THE **6502/6809** JOURNAL

***For fast service call toll free* 1-800-345-8112 (In PA 1-800-662-2444)**

MICRO BOOKS

Quan	Title	Price	s/h	Total
_____	Micro on the Apple Vol. 1	$24.95	$2.00	_____
_____	Micro on the Apple Vol. 2	24.95	2.00	_____
_____	Micro on the Apple Vol. 3	24.95	2.00	_____
_____	What's Where in the Apple...Plus... the All New Guide to What's Where	24.95	2.00	_____
_____	THE GUIDE to WWA	9.95	2.00	_____
	(MA residents add 5% sales tax)		Total	_____

□ PAYMENT ENCLOSED
(Checks payable to MICRO)

□ VISA □ Master Card

Act.# ____________________

Expires ____________________

Name

Address

City

State/Prov.

Zip

Country

Offer good until 10/31/82
Allow 4-6 weeks for delivery

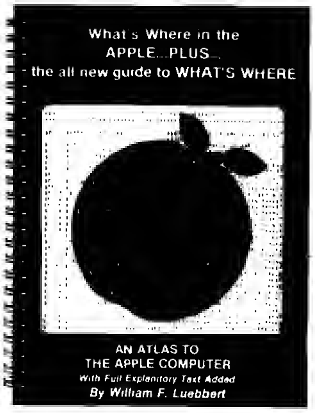
Get The Most Out Of Your APPLE

With The Most Important Book
Ever Published on the APPLE!

What's Where in the APPLE... Plus...the All New Guide to What's Where

Revised Edition of the famous Apple Atlas

- Guides you — with a numerical Atlas and an alphabetical Gazetteer — to over 2,000 memory locations of PEEKs, POKEs, and CALLS.
- Gives names and locations of various Monitor, DOS, Integer BASIC, and Applesoft routines — and tells you how they're used.
- Enables you to move easily between BASIC and Machine Language
- Guides you through the inner workings and hidden mechanisms of the Apple.



All Apple users will find this book helpful in understanding their machine, and essential for mastering it!

The Atlas and THE ALL NEW GUIDE are available in one, 256-page Wire-O-Bound book for only **$24.95**

If you own the original *What's Where in the Apple?* you will want **THE GUIDE** to complement your edition. This 128-page Wire-O-Bound version contains all new material to be used with the memory map and atlas for **$9.95.**

With

MICRO on the APPLE Series

Complete with Diskette

No need to type in hundreds of lines of code.

Vol. 1

Allows you to:

- Round and format numbers accurately
- Get lowercase letters and punctuation into Applesoft strings
- Play the hig game "Spelunker"

Vol. 2

Lets you:

- Speed up machine language programming using 5 powerful machine language aids
- Add additional editing and I/O features
- Play the intriguing game "GalactiCube"

Vol. 3.

Just Released!

Gives you more:

- Machine language aids
- I/O enhancements
- Graphics and games

...and much, much more!

Together Volumes 1, 2, and 3 provide more than 110 programs on diskette for less than $1.00 each.

Volumes 1 and 2: 13 sector DOS 3.2 format.
Volume 3: 16 sector DOS 3.3 format.

Micro on the Apple, Volumes 1, 2, and 3
(Complete with Diskette)

$24.95 each

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD
FIRST CLASS PERMIT NO. 60, CHELMSFORD, MA

POSTAGE WILL BE PAID BY ADDRESSEE

MICRO
34 Chelmsford Street
P.O. Box 6502
Chelmsford, MA 01824

Use Coupon
On Reverse
Side to Order
and
Mail TODAY!

83-356

ROCKWELL Microcomputers from Excert, Inc.

THE AIM 65/40 Single Board or Smorgasbord

- A full size terminal style keyboard w/8 special function keys
- A smart, 40 character display with its own microprocessor
- A 40 column printer w/text and graphic output
- Up to 64K of on-board RAM and ROM
- On-board interfaces include RS232, dual audio cassette and 2 user I/O R6522 devices
- Firmware includes interactive monitor and text editor w/options of Assembler, BASIC, FORTH and PL/65

THE AIM 65 Take-Out Order

- A full size terminal style keyboard w/3 special function keys
- A 20 character display
- A 20 column printer w/text and graphic output capability
- Up to 4K RAM and 20K ROM on-board
- On-board interfaces include 20MA TTY, dual audio cassette and 1 user I/O R6522 device
- Firmware includes interactive monitor and text editor w/options of Assembler, BASIC, FORTH, PASCAL, & PL/65

And if the above isn't enough,
Try the RM65 — a product line filled with embellishments including:

32K DRAM Board
CRT Controller
Floppy Disk Controller
PROM Programmer

ACIA Board
IEEE-488Board
CPU/SBC Board
4-16 Slot Card Cages

Prototype cards
Adaptor Buffer Modules
General Purpose I/O Board
PROM/ROM Board

NEW LOWER PRICES AND A CASH DISCOUNT* TO BOOT!

A65/40-16 (16K RAM)	$1225.
A65/40-32 (32K RAM)	$1295
A65/40-A (Assembler)	$ 85
A65/40-B (BASIC)	$ 65

A65-1 (1K RAM)	$420
A65-4 (4K RAM)	$445
A65-4B (4K RAM w/BASIC)	$495
A65-PS (PASCAL)	$100
A65-F (FORTH)	$ 65
A65-A (Assembler)	$ 35

Mail Order to:

Educational Computer Division
EXCERT INCORPORATED

- SALES
- SERVICE
- INSTALLATION
- CONSULTING

P.O Box 8600
White Bear Lake, MN 55110
(612) 426-4114

Higher quantities quoted upon request, COD's accepted, shipping will be added. *Deduct 5% cash discount on prepaid orders. Minnesota residents add 5% sales tax. Prices subject to change without notice.

Skyles Electric Works

Epson-PET/CBM Graphic ROM Pack

For PET/CBM Owners Who Want:

Complete Program Listing Printouts
Complete Screen Graphic Printouts
Graphic Printouts From Programs

on your Epson Printer

Order the Skyles Electric Works EPSON-PET GRAPHIC ROM Package. The ROM when installed in an EPSON MX80, MX80FT, or MX100 with Graftrax Plus printer will reproduce most of the PET/CBM graphics characters. Most importantly when using the accompanying high speed machine language program, the Epson-PET Graphic ROM pack gives a complete program listing with all screen controls shown (cursor, home, clear, etc.). This high speed machine language program for program listing, screen image printout (screen dumps) and BASIC program controlled printing (i.e. PRINT) automatically translates the PET-ASCII characters to the EPSON-GRAPHIC ROM characters. A BASIC sample program and PRINT subroutine, that may be incorporated into any existing BASIC program, completes this "complete solution" package.

EPSON MODEL	ROM MODEL
MX70	Not Available
MX80 (serial no. to 359999)	EPG80 (EPG82, 3 ROM Version)
MX80 (serial no. after 360000)	EPG81 (EPG83, 3 ROM Version)
MX80FT	EPG8F
MX80 Graftrax	Not Available
MX80 Graftrax Plus	EPG8G+
MX80FT Graftrax Plus	EPG8G+
MX100	Not Available
MX100 Graftrax Plus	EPG10G+

The Epson-PET Graphics ROM Pack has been designed to furnish you with PET/CBM graphics printing in the easiest way possible. This is done by furnishing a high speed machine language program that is "hidden" at the top of your PET/CBM memory.

The machine language program serves 3 major functions.

1: Translates PET-ASCII code to ASCII code for program listing.

2: Translates screen code to ASCII code for screen image printouts.

3: Translates PET-ASCII code strings to ASCII strings for normal program printout. This feature may also be used for making ASCII files for your disk or tape recorder.

INSTALLATION: **Installs into your Epson printer**

PRICE: **Epson-PET Graphics ROM Pack EPG80, EPG81, EPG8F.....$75.00**

Please specify your Epson printer model type and and serial number when ordering.

For all PET/CBM's BASIC 2.0/Revision 3, or BASIC 4.0

AVAILABILITY: Immediately from your LOCAL DEALER

or

VISA, MASTERCHARGE ORDERS CALL (800) 227-9998 (except California residents)
CALIFORNIA ORDERS PLEASE CALL (415) 965-1735

Skyles Electric Works

231E South Whisman Road
Mountain View, CA 94041
(415) 965-1735

Superimposing TV Pictures on PET Video

by Peter D. Hiscocks

The Commodore PET may be modified to superimpose the picture from a television camera on the PET video display. The resultant combined picture (PET video and television image) may also be fed to a remote monitor or video tape recorder.

I had to superimpose the picture from a television camera on the PET video display as part of an airborne camera control unit (see reference). The same concept could be useful in a variety of situations. For example, the composite video signal — television picture plus PET graphics and alphanumerics — could be fed to an extension monitor or video tape recorder and the PET would become a very sophisticated video titling system. In another situation, the signal from a microscope-mounted television camera could be fed to the PET and the area of cells determined by tracing with a light pen.

Note that the television picture is not available to the PET; it is simply overlaid on the display, and the PET doesn't know of its existence. A block diagram of the system is shown in figure 1.

A first requirement of the system is that the television camera scan rate be synchronized to that of the PET display. Fortunately, the vertical and horizontal sync signals are available at the PET User Port. These are modified to a suitable form by the sync circuits shown in figure 2, and fed to the television camera. The television camera must be able to accept and lock on these signals, but this is a common requirement of cameras that are used in studio situations. I used the Sanyo 1620 X camera.

The next requirement of the system is that the display be able to show a continuous grey scale from full dark to full brightness (or greenness, in the case of the new PETs). The PET display, of course, only has to deal with the two extremes of brightness. The grey scale is dealt with by the video amplifier (figure 4) and the modifications to the PET video display (figure 3).

In the existing PET display circuitry, the video amplifier switches the cathode of the CRT between 35 volts (dark level) and ground (white level). The added video amplifier controls the power supply rail of the existing amplifier, thereby setting the maximum dark level and creating a continuous grey scale in response to the video signal. PET characters (white) are visible as long as the video picture is not completely white.

Modifying the PET video display is not for the faint-hearted or the novice electronic tinkerer. There are high

***Figure 1:* Block Diagram**

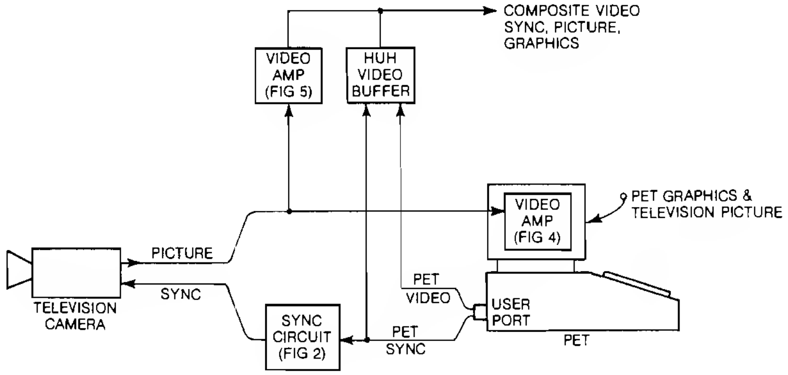


***Figure 2:* Sync Circuits**

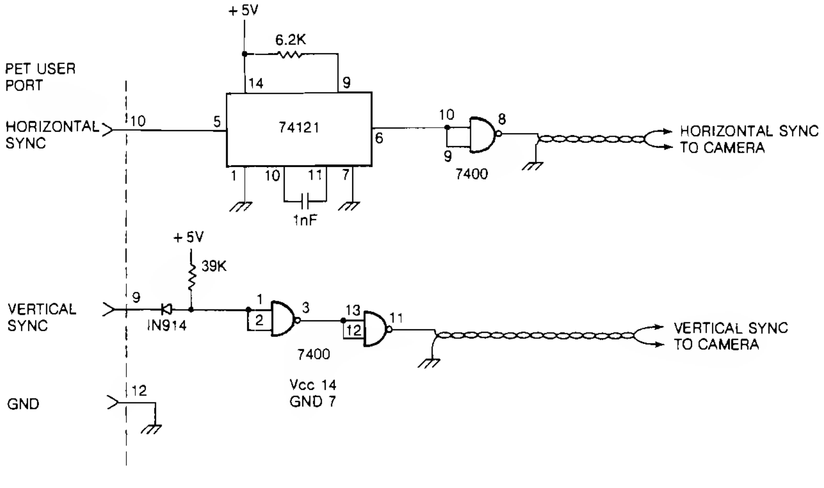


Figure 3: PET Video Amplifier Modifications

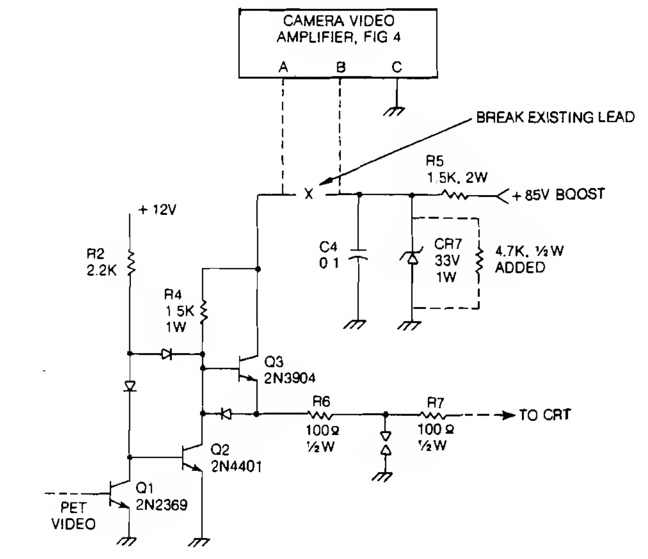


Figure 4: Camera Video Amplifier

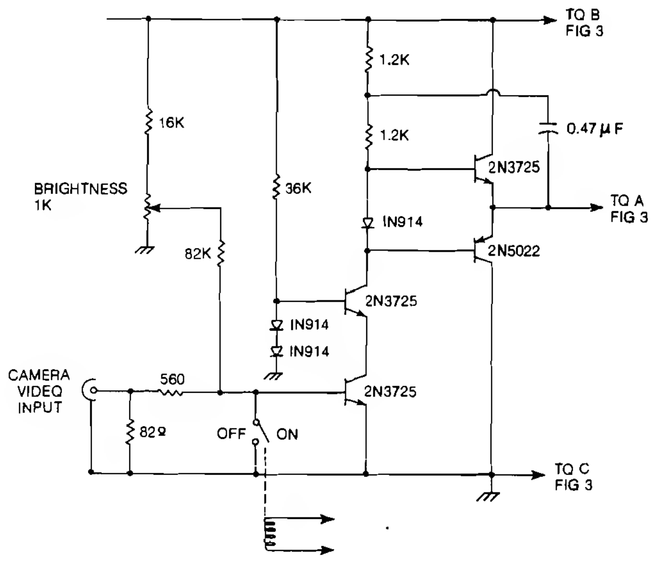
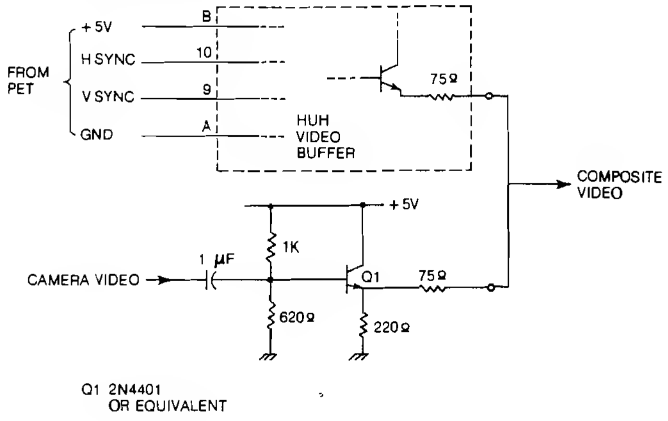


Figure 5: Video Amplifer



voltages inside the display, and mistake could fry you and/or you PET. The circuit details that I show i figure 3 apply to the early PETs an you'd be wise to confirm the circuitry i you have a newer version.

The circuitry of the added video amplifier should be copied fairly closely especially in the choice of transistor (the bandwidth requirement for th amplifier is in the order of 3.5 MHz). made up the circuit on Veroboard an mounted it inside the case of the PE display. In my installation, I used relay to kill the television picture s the toggle switch could be placed o the front panel, without having to ru signal-carrying leads up to the fron panel. Leads from the added vide amplifier to the existing vide amplifier should be kept as short a possible and well away from th display flyback transformer area.

In figure 3, the addition of the 4.7 Kohm resistor prevents excessiv power dissipation in the 33-volt zene diode.

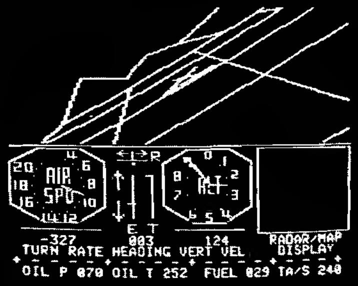
Figure 1 also shows how the combined picture and graphics image ma be sent to some external device such a an external video display or video tap recorder. The video amplifier of figure 5 buffers the video signal from the television camera and the HUH Video Buffe (HUH Electronics, 1429 San Mateo, CA 94402) develops a composite sync signal from the PET sync signals. A alternative suitable circuit for the latte is given in *The PET Revealed*, by Nick Hampshire, page 86.

Reference

"A Computer-Based Camera Control System," by A. Roberts and P. Hiscocks, *Photogrammetric Engineering and Remote Sensing*, Vol. 47, No. 1, Jan. 1981, pp. 53-57.

Peter Hiscocks is an instructor at Ryerson Polytechnical Institute, where he teaches courses in Electrical and Theatre technology. He is a columnist for Audio/Video Canada, builds computer interfaces on a freelance basis, and is currently working on a computer-controlled sound system for the reproduction of insect sounds at the Royal Ontario Museum. He may be contacted at Electrical Dept., Ryerson Polytechnical Institute, 50 Gould St., Toronto, Canada M5B 1E8.

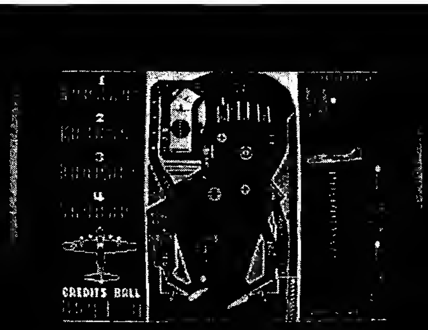
from SubLOGIC...
quality software for your Apple II.



A2-FS1

FLIGHT SIMULATOR

Combines superior flight simulation with the best animated 3D graphics available. Practice take-offs and landing, other aerial maneuvers, declare war on the enemy. 16K cassette, **$25.00**. 32K disk, **$33.50**.



A2-PB1

PINBALL – The ultimate arcade simulation program, an exciting pinball game with the ball and flipper precision to make increased skill pay off. Includes 10 different play modes and 100 user-adjustable modes. 48K disk, **$29.95.**

A2-SG1

ESCAPE! – The simple game of captive pursuit. A gem for game purists . . . A classic for your game collection. 48K disk, **$29.95.**

See them today at your dealer . . .

or for direct orders add $1.50 and specify UPS or first class mail. Illinois residents add 5% sales tax. VISA and MasterCard accepted. Descriptive brochures of most products listed here are available on request

"Apple" is the registered trademark of Apple Computer Inc.

subLOGIC
Communications Corp.
713 Edgebrook Drive
Champaign, IL 61820
(217) 359-8482
Telex: 206995

A/D Conversion Using a 555 Timer IC

by Mike Dougherty

For applications not requiring high-speed analog to digital conversion, this simple 555 Timer circuit yields a high dynamic range for very low cost. Although demonstrated with an Atari 800, this 555 A/D converter may be used by any computer system with one unused bit on an input port.

These programs require:

- Atari 800
- 555 Timer IC
- and other electronic components

The Atari 800 already contains hardware capable of digitizing a resistive sensor, such as a thermistor or photoresistor, through a game controller jack. During each video scan, the resistance of each paddle is measured by a custom-designed LSI chip called POKEY. At the end of a video scan, the Atari Operating System reads the POKEY registers and places the digitized values in the paddle shadow registers in memory. These values are accessed through the PADDLE( ) function of BASIC.

Since the POKEY digitization is done by hardware and the shadow registers are maintained by the operating system, this analog to digital conversion method is easy to use. Simply replace the paddle controller with the desired resistive sensor and obtain the digitized result *via* the proper PADDLE( ) function. The actual A/D conversion will be performed automatically by the hardware and updated by the operating system every 1/60th of a second.

As straightforward as this approach seems, there are two major disadvantages. First, the Atari 800 hardware was designed for the 1MΩ potentiometer used in the paddle controller. To fully utilize the Atari hardware, the variable resistance to be measured should cover this range. Often it is difficult for the hobbyist to find a resistive sensor with this full range. Secondly, even if such a resistive sensor is found, the Atari hardware can only resolve one part in 228. This yields an effective A/D conversion of $LOG_2(228) = 7.8$ bits. There are many applications which could use finer resolution or greater dynamic range.

This article discusses the use of an inexpensive 555 Timer circuit, the software required to digitize the 555 output, and software designed to display the 555 Timer waveform on the Atari 800. Although the Atari 800 is used to demonstrate the 555 Timer circuit, the general principles may be applied to any 6502-based computer.

Hardware

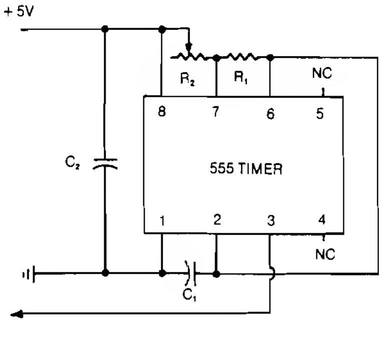
The 555 Timer is available from Radio Shack in an eight-pin DIP package for approximately $1 (Radio Shack part number 276-1723). This application uses the 555 as a free-running astable multivibrator, shown in the circuit of figure 1. In this mode of operation, the 555 outputs a TTL-compatible pulse stream on pin 3. The waveform of the pulse stream is determined by the external resistors, R_1 and R_2, and the capacitor, C_1. For each single pulse, pin 3 remains at a low voltage for a time, t_L, proportional to R_1. Pin 3 remains at a high voltage for a time, t_H, proportional to $R_1 + R_2$. Specifically, these times are defined:

$$t_H = 0.693\,(R_1 + R_2)\,(C_1) \text{ seconds}$$

$$t_L = 0.693\,(R_1)\,(C_1) \text{ seconds}$$

Thus, as the resistive sensor R_2 varies, the time that the pulse is high also varies.

Figure 1: **555 Timer Circuit**



555 PULSE STREAM NC: NO CONNECTION

By attaching 555 pin 3 to an input pin on a parallel port, the length of time that the pulse is high may be measured by computer clock speed. In the astable mode, only a single input line is required to digitize R_2.

The circuit of figure 1 was powered through the joystick #4 controller jack. Pin 7 of joystick #4 supplies the +5 volt source, while pin 8 is the Atari ground. With the Atari 800 providing the +5V power source through the controller jack, a random variability up to 10% was observed. This error was eliminated with the addition of a filtering capacitor, C_2, across the +5V source and ground.

Software

One simple approach to measuring the high portion of the pulse is outlined by the "pseudocode" in listing 1.

The first two DOWHILE loops allow the pseudocode to synchronize with the free-running 555 pulse stream. In the worst case, this synchronization will take $0.693(2R_1 + R_2)(C_1)$ seconds. Because of this relationship, R_2 was chosen as the resistive sensor and R_1 was made as small as possible. Thus the time that a pulse is low is kept constant by R_1 and is only long enough for the first and third DOWHILE loops to recognize the zero level of the 555 output.

The time represented by each count is determined by the length of the third

DOWHILE loop and the specific delay utilized. Typically, the exact length of the third DOWHILE loop is not important since most applications will require a series of calibrations to map the digitized counts onto the physical quantity being measured. In practice, the third DOWHILE loop is written to execute as fast as possible. If this is not fast enough, then the value of capacitor C_1 is increased. By controlling both the measuring loop and the capacitor value of C_1, a wide range of resistive sensors may be utilized.

As an example, the 6502 assembler subroutine in listing 2 demonstrates a 19-cycle timing loop. With a 1 MHz clock, this loop will sample the input line, bit 0 of PORT, every 19 microseconds. Assume that C_1 is 0.1 μF, R_1 is fixed at 2KΩ, and the resistive sensor covers the range from 2KΩ to 700KΩ (the practical range of the paddle game controllers). Then t_H will vary from 0.2777 ms at 2KΩ to 48.6 ms at 700KΩ with the digitized count varying from COUNT = 15 to COUNT = 2558, a range of 2543 counts. The worst case time to synchronize and take the sample at the maximum resistance is $2t_H + t_L$ = 97.5 ms, or approximately 1/10th of a second.

Listing 1: Pseudocode A/D Conversion

```
BEGIN Pulse;
  initialize COUNT to zero;
  DOWHILE line is not equal to zero;
    read I/O port;
    logically "AND" with MASK to isolate 555 line;
  ENDWHILE;
  DOWHILE line is equal to zero;
    read I/O port;
    logically "AND" with MASK to isolate 555 line;
  ENDWHILE;
  DOWHILE line is not equal to zero;
    increment COUNT by one;
    delay;
    read I/O port;
    logically "AND" with mask to isolate 555 line;
  ENDWHILE;
END Pulse.
```

Listing 2: Sample 6502 A/D Conversion Subroutine

```
              0100 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
              0110 ;
              0120 ; LISTING # 2
              0130 ;
              0140 ; SAMPLE 6502 SUBROUTINE TO COUNT
              0150 ; THE NUMBER OF LOOPS THAT BIT0 OF
              0160 ; I/O PORT IS HIGH.  THE RESULT IS
              0170 ; LEFT IN ZERO PAGE MEMORY.  THE
              0180 ; COUNTING LOOP CONTAINS "DELAY"
              0190 ; INSTRUCTIONS TO MAKE THE LOOP
              0200 ; EXECUTION TIME 19 CYCLES.
              0210 ;
              0220 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
              0230 ;
              0240 ;
D301          0250 PORT   =    $D301      PORTB OF ATARI
00D4          0260 COUNT  =    $D4        ZERO PAGE RESULT BUFFER
              0270 ;
0000          0280        *=   $0600      ATARI FREE MEMORY
              0290 ;
0600 A900     0300 PULSE  LDA  #$00       INIT Y,X AS COUNT
0602 AA       0310        TAX             THIS IS FASTER
0603 A8       0320        TAY             THAN MEMORY
              0330 ;
              0340 ; SYNCHRONIZE WITH 555 PULSE STREAM
              0350 ;
0604 AD01D3   0360 SYNCHI LDA  PORT       INPUT ALL 8 LINES
0607 2901     0370        AND  #$01       MASK TO ONLY BIT0
0609 D0F9     0380        BNE  SYNCHI     STILL HIGH, WAIT TILL LOW
              0390 ;
060B AD01D3   0400 SYNCLO LDA  PORT       INPUT ALL 8 LINES
060E 2901     0410        AND  #$01       MASK TO ONLY BIT0
0610 F0F9     0420        BEQ  SYNCLO     STILL LOW, WAIT TILL HIGH
              0430 ;
              0440 ; COUNT # LOOPS PULSE IS HIGH
              0450 ;
0612 E8       0460 CNTLP  INX             COUNT THIS TIME IN LOOP
0613 D00C     0470        BNE  NOCARY     NO CARRY TO MSB
0615 C8       0480        INY             ADD CARRY TO MSB
0616 EA       0490        NOP             DELAY FOR 4 CYCLES
0617 EA       0500        NOP
0618 AD01D3   0510        LDA  PORT       INPUT I/O PORT
061B 2901     0520        AND  #$01       ISOLATE BIT0
061D D0F3     0530        BNE  CNTLP      STILL HIGH -- COUNT ON
061F F00A     0540        BEQ  RETURN     WRAP IT UP
              0550 ;
0621 EA       0560 NOCARY NOP             DELAY FOR 5 CYCLES
0622 A5FF     0570        LDA  $FF
0624 AD01D3   0580        LDA  PORT       INPUT I/O PORT
0627 2901     0590        AND  #$01       ISOLATE BIT0
0629 D0E7     0600        BNE  CNTLP      STILL HIGH -- COUNT ON
              0610 ;
              0620 ; RESULT IN Y,X REGISTER PAIR
              0630 ;
062B 86D4     0640 RETURN STX  COUNT      SAVE IN ZERO PAGE
062D 84D5     0650        STY  COUNT+1
062F 60       0660        RTS
0630          0670        .END
```

If a longer delay may be tolerated, then C_1 may be increased to 1μF. In this case, t_H will vary from 2.77 ms to 486 ms with the digitized count varying from 146 to 25579, a range of 25433 counts. Thus the 0.1 μF capacitor gives a digitized dynamic range of $LOG_2(2543)$ = 11.3 bits; the 1.0 μF capacitor gives a dynamic range of $LOG_2(25433)$ = 14.6 bits. If time is not a critical factor, the 555 approach allows high dynamic range for very low cost.

Atari 800 Considerations

The approach outlined in the previous sections may be implemented on the Atari after dealing with two operating system problems. Although these problems are specific to the Atari, other computer systems may have similar difficulties.

The Atari computer system drives the video display through direct memory access (DMA). A sophisticated approach utilizing the custom-designed LSI chips, called ANTIC and CTIA, provides the graphics versatility of the Atari. To optimize the video performance, the Atari has a basic clock speed of 1.79 MHz rather than the more common 1 MHz. However, the actual speed of program execution depends upon the amount of "cycle stealing" DMA activity required to run the video screen. As the graphics mode uses more memory to define the video screen, more DMA data transfers are required to refresh the screen. Thus a program will execute slower under GRAPHICS 8 mode than under GRAPHICS 0 mode. In addition, for graphics modes combined with text (e.g. GRAPHICS 8), the DMA rate will differ between displaying the graphics portion and the text portion of the screen. This variable rate DMA affects the effective speed of the counting loop used to measure the 555 pulse width.

Fortunately, there is a simple solution to the DMA problem — turn the DMA off. You can disable the DMA by writing a zero in the DMA shadow register, SDMCTL, located at $022F in memory. This value is transferred to the DMA controller register by the operating system during the VBLANK interrupt routine, every 1/60th of a second. (VBLANK is the interrupt routine executed after every complete video scan. The Atari Operating System uses VBLANK to update the system time, input the game controller reigsters, transfer shadow registers to and from hardware, and other housekeeping tasks.) Thus to turn off the DMA, simply POKE a zero into memory location $022F and wait for the system clock to change (low order byte at $0014). Since VBLANK updates the clock, the change in the system clock value indicates that VBLANK has executed and the DMA shadow register has been transferred to the DMA hardware. To restore the DMA, POKE a $22 (decimal 34) to the DMA shadow register at memory location $022F.

The second problem concerns the actual time taken to execute VBLANK. A typical 555 digitization time was about a tenth of a second or longer. During this digitization, VBLANK could interrupt several times. Since processor time would be spent executing the VBLANK routine instead of counting, the digitized value would be too low. In addition, the short low period of the pulse could occur during VBLANK execution and be missed by the counting subroutine. The counting subroutine would erroneously continue until a low pulse was recognized.

The Atari designers anticipated this type of problem also. Although the VBLANK interrupt is connected to the 6502 non-maskable interrupt line, NMI, external hardware was provided to allow an Atari program to disable this NMI interrupt before it reaches the

***Listing 3:* Atari USR Function A/D Conversion**

```
                 0100 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                 0105 ;
                 0110 ; LISTING # 3
                 0115 ;
                 0120 ; ATARI USR FUNCTION TO COUNT THE
                 0125 ; LOOPS WHICH A 555 PULSE IS HIGH.
                 0130 ; THE STEPS PERFORMED ARE:
                 0135 ;
                 0140 ; 1) TURN OFF DMA SHADOW REGISTER
                 0145 ; 2) WAIT FOR VBLANK TO TRANSFER
                 0150 ;    THE SHADOW REG TO HARDWARE
                 0155 ; 3) DISABLE THE VBLANK INTERRUPT
                 0160 ; 4) SYNCHRONIZE WITH THE 555
                 0165 ;    PULSE STREAM WITH 2 LOOPS
                 0170 ; 5) COUNT THE NUMBER OF LOOPS
                 0175 ;    THE PULSE IS HIGH
                 0180 ; 6) SAVE THE RESULTS IN THE ZERO
                 0185 ;    PAGE USR RETURN LOCATIONS,
                 0190 ;    RESET VBLANK AND DMA
                 0195 ;
                 0200 ;
                 0205 ; USAGE:  COUNT=USR(1696,LINE)
                 0210 ;
                 0215 ;    WHERE LINE IS THE MASK TO
                 0220 ;    ISOLATE THE BIT CONNECTED
                 0225 ;    TO THE 555 PULSE STREAM.
                 0230 ;    UPON RETURN, COUNT WILL
                 0235 ;    CONTAIN THE DIGITIZED VALUE.
                 0240 ;
                 0245 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                 0250 ;
                 0255 ;
022F             0260 DMASR  =      $022F      DMA SHADOW REGISTER
D40E             0265 NMIEN  =      $D40E      NMI ENABLE REGISTER
D301             0270 PORT   =      $D301      PORTB -- JOYSTICK #3,#4
0014             0275 RTC    =      $0014      LSB OF REAL TIME CLOCK
00D4             0280 USRRET =      $00D4      USR RETURN ADDR
00D1             0285 LINE   =      $00D1      STORAGE FOR LINE MASK
                 0290 ;
                 0295 ;
0000             0300        *=     $06A0      FREE MEMORY IN PAGE 6
                 0305 ;
06A0 68          0310 ATOD   PLA               # OF USR ARGUMENTS
06A1 68          0315        PLA               HIGH ORDER OF LINE
06A2 68          0320        PLA               LOW ORDER OF LINE
06A3 85D1        0325        STA    LINE       SAVE 8 BIT LINE MASK
                 0330 ;
                 0335 ; TURN OFF DMA SHADOW REGISTER
                 0340 ;
06A5 A900        0345        LDA    #$00
06A7 8D2F02      0350        STA    DMASR      CLEAR SHADOW REGISTER
06AA AA          0355        TAX               INITIALIZE REGS FOR COUNT
06AB A8          0360        TAY
                 0365 ;
                 0370 ; WAIT FOR VBLANK INTERRUPT
                 0375 ;
06AC A514        0380        LDA    RTC        CURRENT LSB OF RTC
06AE C514        0385 RTCWAT CMP    RTC        SEE IF RTC HAS CHANGED
06B0 F0FC        0390        BEQ    RTCWAT     NOT YET, WAIT
                 0395 ;
                 0400 ; DISABLE VBLANK INTERRUPT
                 0405 ;
06B2 A920        0410        LDA    #$20
06B4 8D0ED4      0415        STA    NMIEN      CLEAR BIT6 OF NMIEN
                 0420 ;
                 0425 ; SYNCHRONIZE TO 555 PULSE STREAM
                 0430 ;
06B7 AD01D3      0435 SYNCHI LDA    PORT       INPUT ALL 8 LINES
06BA 25D1        0440        AND    LINE       ISOLATE 555 BIT
06BC D0F9        0445        BNE    SYNCHI     STILL HIGH, WAIT
                 0450 ;
06BE AD01D3      0455 SYNCLO LDA    PORT       INPUT ALL 8 LINES
06C1 25D1        0460        AND    LINE       ISOLATE 555 BIT
06C3 F0F9        0465        BEQ    SYNCLO     STILL LOW, WAIT
                 0470 ;
                 0475 ; DIGITIZE TIME THAT PULSE IS HIGH
                 0480 ;
06C5 E8          0485 CNTLP  INX               COUNT THIS TIME IN LOOP
06C6 D00C        0490        BNE    NOCARY     NO CARRY TO MSB OF COUNT
06C8 C8          0495        INY               CARRY TO MSB OF COUNT
06C9 A5FF        0500        LDA    $FF        DELAY 3 CYCLES
```

(Continued)

Listing 3 *(Continued)*

```
06CB AD01D3 0505         LDA   PORT       INPUT I/O PORT
06CE 25D1   0510         AND   LINE       ISOLATE 555 BIT
06D0 D0F3   0515         BNE   CNTLP      STILL HIGH -- COUNT ON
06D2 F009   0520         BEQ   RETURN     LOW, WRAP IT UP
            0525 ;
06D4 EA     0530 NOCARY  NOP              DELAY 4 CYCLES
06D5 EA     0535         NOP
06D6 AD01D3 0540         LDA   PORT       INPUT I/O PORT
06D9 25D1   0545         AND   LINE       ISOLATE 555 BIT
06DB D0E8   0550         BNE   CNTLP      STILL HIGH -- COUNT ON
            0555 ;
            0560 ; SAVE RESULTS, RESTORE HARDWARE
            0565 ;
06DD 86D4   0570 RETURN  STX   USRRET     USR RESULT BUFFER
06DF 84D5   0575         STY   USRRET+1
06E1 A960   0580         LDA   #$60       SET BIT6 OF NMIEN
06E3 8D0ED4 0585         STA   NMIEN      TO ENABLE VBLANK AGAIN
06E6 A922   0590         LDA   #$22
06E8 8D2F02 0595         STA   DMASR      ENABLE DMA THRU VBLANK
06EB 60     0600         RTS              BACK TO BASIC
06EC        0605         .END
```

Listing 4: **USR Function to Trace Atari Port B**

```
            0100 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            0105 ;
            0110 ; LISTING # 4
            0115 ;
            0120 ; ATARI USR FUNCTION TO SAMPLE
            0125 ; AND STORE THE VALUE OF PORTB,
            0130 ; JOYSTICK #3 AND #4.  THE STEPS
            0135 ; PERFORMED ARE:
            0140 ;
            0145 ; 1) INITIALIZE
            0150 ; 2) DISABLE DMA AND VBLANK
            0155 ; 3) IN 256 BYTE BLOCKS, SAMPLE
            0160 ;    AND STORE PORTB
            0165 ; 4) RESTORE VBLANK AND DMA
            0170 ;
            0175 ;
            0180 ; USAGE:  X=USR(1536,ADR(DATA$))
            0185 ;
            0190 ; WHERE DATA$ IS A STRING VARIABLE
            0195 ;    DIMENSIONED TO BE 1024 BYTES
            0200 ;    IN SIZE TO HOLD THE ACTUAL
            0205 ;    PORTB VALUES.
            0210 ;
            0215 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            0220 ;
            0225 ;
00CB        0230 ADDR   =    $00CB     STORAGE FOR DATA ADDR
D301        0235 PORT   =    $D301     ATARI PORTB
0014        0240 RTC    =    $0014     LSB OF REAL TIME CLOCK
022F        0245 DMASR  =    $022F     DMA SHADOW REGISTER
D40E        0250 NMIEN  =    $D40E     NMI ENABLE REGISTER
            0255 ;
            0260 ;
0000        0265        *=   $0600     FREE ATARI RAM
            0270 ;
            0275 ; INITIALIZE TRACE
            0280 ;
0600 68     0285 TRACE  PLA            # OF USR ARGUMENTS
0601 68     0290        PLA            16 BIT DATA BUFFER ADDR
0602 85CC   0295        STA  ADDR+1    MSB OF DATA$
0604 68     0300        PLA
0605 85CB   0305        STA  ADDR      LSB OF DATA$
0607 A000   0310        LDY  #$00      INDIRECT INDEXED POINTER
0609 A204   0315        LDX  #$04      FOUR BLOCKS OF 256 SAMPLES
            0320 ;
            0325 ; DISABLE DMA AND VBLANK
            0330 ;
060B A900   0335        LDA  #$00
060D 8D2F02 0340        STA  DMASR     TURN OFF SHADOW REGISTER
0610 A514   0345        LDA  RTC       WAIT UNTIL VBLANK INTERRUPT
0612 C514   0350 RTCWAT CMP  RTC       NEW RTC VALUE ?
0614 F0FC   0355        BEQ  RTCWAT    NOT YET, WAIT
0616 A920   0360        LDA  #$20      YES, DISABLE VBLANK
0618 8D0ED4 0365        STA  NMIEN     CLEAR BIT6 OF NMIEN
            0370 ;
            0375 ; TAKE TRACE IN 256 BYTE BLOCKS
            0380 ;
```

(Continued)

Table 1: **Atari 800 PORTB Pinouts**

PORTB Bit	Joystick	Pin#	Mask
0	#3	1	1
1	#3	2	2
2	#3	3	4
3	#3	4	8
4	#4	1	16
5	#4	2	32
6	#4	3	64
7	#4	4	128

6502 chip. This interrupt line is controlled by bit 6 of the hardware register NMIEN at location $D40E. Clearing bit 6 of NMIEN disables the VBLANK interrupt; setting bit 6 of NMIEN enables the VBLANK interrupt. A word of caution: this register also controls other interrupts. To disable the VBLANK interrupt, POKE a $20 into NMIEN; to enable the VBLANK interrupt, POKE a $60 into NMIEN.

A BASIC callable USR function to sample an input pin of PORTB, joystick #3 and #4, is given in listing 3. The USR argument is used to mask the 8-bit I/O port to the single I/O line attached to the 555 pulse stream of pin 3. The pinouts for PORTB and associated masks are outlined in table 1.

TRACE

To test the 555 circuit, a simple program was written to trace the PORTB input lines and store the results in memory to be later plotted on the screen. This program, TRACE, takes a trace of 1024 samples and plots the results in GRAPHICS 8 mode *via* the two USR functions in listings 4 and 5. TRACE also uses the USR function of listing 3 to allow the pulse stream to be digitized. The BASIC program is presented in listing 6.

TRACE allows any input pin of PORTB to be sampled and plotted. The specific line is determined by the value of the bit mask variable, LINEMASK. Refer to table 1 for the PORTB pinouts and corresponding mask values.

The loops used to perform the A/D conversion and to trace the waveform take 19 cycles. However, TRACE does contain one timing error. The trace function breaks the 1024 samples into four blocks of 256 single byte samples. This is because the 6502 indirect indexed addressing mode handles 256-byte blocks quickly and conveniently. The extra clock cycles between each 256-byte block were preferred over slowing down the basic loop simply to sample all trace points at equal time intervals.

The plot of the trace is done by a USR function simply to make TRACE more convenient to use. The short marks displayed to the left of each 256-byte trace segment indicate logic 1 and 0.

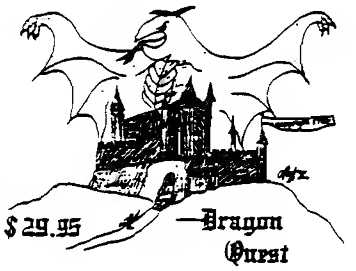
These techniques form the basis for measuring physical quantities through resistive sensors inexpensively. With very simple circuits much of the world may be sensed by your computer.

References

1. Lancaster, Don, "TTL Cookbook," Howard W. Sams & Co., Inc., Indianapolis, Indiana. Copyright 1978.

2. Mims III, Forrest M., "Engineer's Notebook, Integrated Circuit Applications," Radio Shack, USA. Copyright 1979.

Mike Dougherty currently works at Martin Marietta Aerospace in Denver, Colorado. His home-based system presently consists of an Atari 800 with 24K bytes of memory, the Atari 410 recorder, and the Atari 850 Interface Module for future communication with single board computers. Address correspondence to 7659 West Fremont Ave., Littleton, CO 80123.



Listing 4 *(Continued)*

```
061B EA     0385 COLLCT NOP               LOOP MUST BE 19 CYCLES
061C EA     0390        NOP               TO MATCH A/D CONVERSION
061D AD01D3 0395        LDA   PORT        SAMPLE PORTB
0620 91CB   0400        STA   (ADDR),Y    SAVE IN DATA$
0622 C8     0405        INY               NEXT MEMORY LOCATION
0623 D0F6   0410        BNE   COLLCT      FOR 256 BYTES
0625 E6CC   0415        INC   ADDR+1      POINT TO NEXT MEM PAGE
0627 CA     0420        DEX               ALL FOUR BLOCKS ?
0628 D0F1   0425        BNE   COLLCT      NO, SAMPLE NEXT BLOCK
            0430 ;
            0435 ; RESTORE ATARI VBLANK AND DMA
            0440 ;
062A A960   0445        LDA   #$60        RESET BIT6 OF NMIEN
062C 8D0ED4 0450        STA   NMIEN       ALLOW VBLANK INTERRUPT
062F A922   0455        LDA   #$22        SET DMA SHADOW REGISTER
0631 8D2F02 0460        STA   DMASR       LEET VBLANK SET HARDWARE
0634 60     0465        RTS               BACK TO BASIC
0635        0470        .END
```

***Listing 5:* USR Function to Plot Trace Data**

```
             0100 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
             0105 ;
             0110 ; LISTING # 5
             0115 ;
             0120 ; ATARI USR FUNCTION TO PLOT A
             0125 ; 256 BYTE LOGIC GRAPH FROM BIT
             0130 ; NUMBER "LINE" OF THE DATA.
             0135 ;
             0140 ; USAGE:
             0145 ; X=USR(1600,data,one,zero,line)
             0150 ;
             0155 ; WHERE:
             0160 ; data IS THE ADDRESS OF THE 256
             0165 ;      BYTES OF DATA TO BE PLOTTED
             0170 ; one  IS THE LEFT HAND ADDRESS OF
             0175 ;      THE LOGIC HIGH LINE
             0180 ; zero IS THE LEFT HAND ADDRESS OF
             0185 ;      THE LOGIC LOW LINE
             0190 ; line IS THE MASK TO ISOLATE THE
             0195 ;      SPECIFIC BIT OF data TO BE
             0200 ;      PLOTTED
             0205 ;
             0210 ;
             0215 ; THE FOLLOWING STEPS ARE PERFORMED:
             0220 ;
             0225 ; 1) INITIALIZE
             0230 ; 2) FOR EACH BYTE OF DATA:
             0235 ;      ISOLATE BIT "LINE"
             0240 ;      IF BIT=0 THEN
             0245 ;         SET BIT IN ONE LINE
             0250 ;      ELSE
             0255 ;         SET BIT IN ZERO LINE
             0260 ;      ENDIF
             0265 ;      ROTATE THE MASK PATTERN RIGHT
             0270 ;      IF ALL 8 BITS THEN
             0275 ;         RESET THE MASK
             0280 ;         INCREMENT THE GRAPH INDEX
             0285 ;      ENDIF
             0290 ;
             0295 ; NOTE: DUE TO THE WAY THE GRAPHICS
             0300 ; SCREEN IS ADDRESSED, THE LABLES
             0305 ; ONE AND ZERO SEEM BACKWARD.
             0310 ;
             0315 ;
             0320 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
             0325 ;
             0330 ;
00CB         0335 DATA   =    $00CB     FOR DATA ADDR STORAGE
00CD         0340 ONE    =    $00CD     FOR LOGIC HIGH ADDR STORAGE
00CF         0345 ZERO   =    $00CF     FOR LOGIC LOW ADDR STORAGE
00D1         0350 LINE   =    $00D1     BIT MASK
06FF         0355 DINDEX =    $06FF     DATA INDEX
06FE         0360 GINDEX =    $06FE     GRAPH INDEX
06FD         0365 MASK   =    $06FD     TO SET BIT IN GRAPH
             0370 ;
             0375 ;
0000         0380        *=   $0640     FREE ATARI RAM
             0385 ;
             0390 ; INITIALIZE
             0395 ;
```

(Continued)

Listing 5 *(Continued)*

```
0640 68       0400 PLOT    PLA                  # OF USR ARGUMENTS
0641 68       0405         PLA                  GET DATA ADDRESS
0642 85CC     0410         STA    DATA+1        MSB
0644 68       0415         PLA
0645 85CB     0420         STA    DATA          LSB
0647 68       0425         PLA                  GET ONE LINE ADDRESS
0648 85CE     0430         STA    ONE+1         MSB
064A 68       0435         PLA
064B 85CD     0440         STA    ONE           LSB
064D 68       0445         PLA                  GET ZERO LINE ADDRESS
064E 85D0     0450         STA    ZERO+1        MSB
0650 68       0455         PLA
0651 85CF     0460         STA    ZERO          LSB
0653 68       0465         PLA                  GET PORTB LINE MASK
0654 68       0470         PLA                  ONLY 8 BIT VALUE
0655 85D1     0475         STA    LINE          LSB
0657 A900     0480         LDA    #$00
0659 8DFF06   0485         STA    DINDEX        INIT DATA INDEX
065C 8DFE06   0490         STA    GINDEX        INIT GRAPH INDEX
065F A901     0495         LDA    #$01
0661 8DFD06   0500         STA    MASK          GRAPHICS BIT POSITION
              0505 ;
              0510 ; PLOT EACH BYTE OF THE 256 BYTE
              0515 ; BLOCK OF DATA
              0520 ;
0664 ACFF06   0525 PLOTLP  LDY    DINDEX        DATA INDEX
0667 B1CB     0530         LDA    (DATA),Y      GET BYTE OF DATA
0669 25D1     0535         AND    LINE          ISOLATE TO SINGLE BIT
066B D00C     0540         BNE    NEO           PLOT <> 0 VLAUE
              0545 ;
066D ACFE06   0550         LDY    GINDEX        GRAPH INDEX
0670 ADFD06   0555         LDA    MASK          CURRENT BIT POSITION
0673 11CD     0560         ORA    (ONE),Y       SET BIT
0675 91CD     0565         STA    (ONE),Y       REPLACE IN GRAPH
0677 D00A     0570         BNE    NEXT          BRANCH ALWAYS
              0575 ;
0679 ACFE06   0580 NEO     LDY    GINDEX        GRAPH INDEX
067C ADFD06   0585         LDA    MASK          CURRENT BIT POSITION
067F 11CF     0590         ORA    (ZERO),Y      SET BIT
0681 91CF     0595         STA    (ZERO),Y      REPLACE IN GRAPH
              0600 ;
0683 18       0605 NEXT    CLC                  ROTATE MASK THRU CARRY
0684 6EFD06   0610         ROR    MASK          NEXT BIT IN GRAPH
0687 D004     0615         BNE    SKIP
0689 6EFD06   0620         ROR    MASK          ROTATE INTO BIT7
068C C8       0625         INY                  PLOT TO NEXT GRAPH BYTE
068D 8CFE06   0630 SKIP    STY    GINDEX        RESTORE GRAPH INDEX
0690 EEFF06   0635         INC    DINDEX        NEXT BYTE OF DATA
0693 D0CF     0640         BNE    PLOTLP        FOR ALL 256 BYTES
0695 60       0645         RTS                  RETURN TO BASIC
0696          0650         .END
```

Listing 6: **BASIC Program TRACE**

```
1 REM --------- T R A C E ----------
2 REM -
3 REM - Low Resolution Logic Analyser
4 REM - and 555 Timer A/D Converter
5 REM -
6 REM -     by Mike Dougherty
7 REM -
8 REM ------------------------------
9 REM
1110 DIM BYTE$(2),DATA$(1024)
1115 TRAP 2000:REM WHEN ALL USR FUNCTIONS HAVE BEEN POKED
1120 READ ADDR:REM STARTING ADDRESS OF CODE
1130 FOR CODE=0 TO 1 STEP 0
1140 READ BYTE$:REM BYTE OF CODE
1150 IF BYTE$="**" THEN SOUND 0,0,0,0:GOTO 1120:REM GET NEXT
     USR FUNCTION
1160 GOSUB 1400:REM CONVERT BYTE$->BYTE
1170 POKE ADDR,BYTE:REM PLACE IN MEMORY
1175 ADDR=ADDR+1
1180 NEXT CODE
1400 REM
1401 REM ---CONVERT BYTE$ TO BYTE
1402 REM
1410 BYTE=0
1420 V=ASC(BYTE$(1)):GOSUB 1500
1430 V=ASC(BYTE$(2)):GOSUB 1500
1440 RETURN
1500 REM
```

(Continued)

Listing 6 *(Continued)*

```
1510 IF V<58 THEN BYTE=BYTE*16+V-48
1520 IF V>57 THEN BYTE=BYTE*16+V-55
1530 SOUND 0,BYTE,10,8
1540 RETURN
1600 REM
1601 REM ---SAMPLE THE PORT EVERY 19
1602 REM ---CYCLES AND STORE INTO
1603 REM ---THE BYTE ARRAY, DATA$.
1604 REM
1605 DATA 1536
1610 DATA 68,68,85,CC,68,85,CB
1615 DATA A0,00,A2,04,A9,00,8D,2F,02
1617 DATA A5,14,C5,14,F0,FC
1625 DATA A9,20,8D,0E,D4
1630 DATA EA,EA,AD,01,D3,91,CB
1635 DATA C8,D0,F6,E6,CC,CA,D0,F1
1640 DATA A9,60,8D,0E,D4,A9,22
1645 DATA 8D,2F,02,60
1695 DATA **
1700 REM
1701 REM ---PLOT A ROW OF DATA$ ON
1702 REM ---THE SCREEN AS DEFINED
1703 REM ---BY THE ARGUMENTS.
1704 REM
1705 DATA 1600
1710 DATA 68,68,85,CC,68,85,CB
1715 DATA 68,85,CE,68,85,CD
1720 DATA 68,85,D0,68,85,CF
1721 DATA 68,68,85,D1
1722 DATA A9,00,8D,FF,06,8D,FE,06
1725 DATA A9,01,8D,FD,06
1730 DATA AC,FF,06,B1,CB,25,D1
1735 DATA D0,0C,AC,FE,06,AD,FD,06
1740 DATA 11,CD,91,CD,D0,0A
1745 DATA AC,FE,06,AD,FD,06
1747 DATA 11,CF,91,CF
1750 DATA 18,6E,FD,06,D0,04
1752 DATA 6E,FD,06,C8
1755 DATA 8C,FE,06
1757 DATA EE,FF,06,D0,CF,60
1795 DATA **
1800 REM
1801 REM ---SAMPLE A BIT DEFINED BY
1802 REM ---THE ARGUMENT MASK AND
1803 REM ---MEASURE THE TIME THE
1804 REM ---PULSE IS HIGH IN 19 CYCLE
1805 REM ---TIME UNITS.
1806 REM
1810 DATA 1696
1812 DATA 68,68,68,85,D1
1815 DATA A9,00,8D,2F,02,AA,A8
1820 DATA A5,14,C5,14,F0,FC
1825 DATA A9,20,8D,0E,D4
1830 DATA AD,01,D3,25,D1,D0,F9
1835 DATA AD,01,D3,25,D1,F0,F9
1840 DATA E8,D0,0C,C8,A5,FF,AD,01,D3
1845 DATA 25,D1,D0,F3,F0,09
1850 DATA EA,EA,AD,01,D3,25,D1,D0,E8
1855 DATA 86,D4,84,D5,A9,60
1860 DATA 8D,0E,D4,A9,22,8D,2F,02,60
1895 DATA **
2000 REM
2001 REM -REPEAT FOREVER:
2002 REM -  INPUT THE LINE MASK TO
2003 REM -  CONTROL THE LINE SAMPLED;
2004 REM -  SET UP SCREEN FOR TRACE;
2005 REM -  DO UNTIL "ESCAPE" KEY:
2006 REM -    IF KEY = "A":
2007 REM -      PERFORM A/D ON LINE
2008 REM -      DEFINED BY LINEMASK;
2009 REM -      TURN OFF KEY;
2010 REM -    ELSE IF KEY = "C":
2011 REM -      PERFORM A/D ON LINE
2012 REM -      DEFINED BY LINEMASK;
2013 REM -    ELSE IF KEY = "T":
2014 REM -      TURN OFF KEY;
2015 REM -      DO A TRACE OF PORT;
2016 REM -      PLOT BIT DEFINED BY
2017 REM -      LINEMASK;
2018 REM -    ENDIF;
2019 REM -  ENDDO;
2020 REM -END REPEAT.
2021 REM
2022 REM
2110 GRAPHICS 0
```

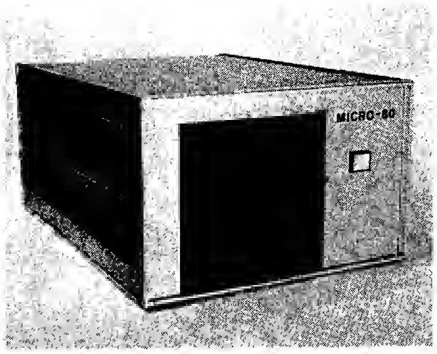
(Continued)

Listing 6 *(Continued)*

```
2120 POSITION 10,10:PRINT "Line mask ";
2130 INPUT LINEMASK
2140 REM
2150 REM
2210 GRAPHICS 8:POKE 752,1
2220 SETCOLOR 1,6,14
2230 SETCOLOR 2,6,4
2240 COLOR 1
2250 REM
2260 REM
2410 FOR LOOP=0 TO 1 STEP 0
2415 PRINT " T - trace   A - A/D "
2420 IF PEEK(764)=63 THEN PRINT " T - trace   A - A/O    ";"A/O: ";
USR(1696,LINE MASK):POKE 764,255
2421 IF PEEK(764)=28 THEN POKE 764,255:GOTO 2000
2422 IF PEEK(764)=18 THEN PRINT " T - trace   A - A/D    ";"A/O: ";
USR(1696,LINE MASK)
2423 IF PEEK(764)<>45 THEN 2420
2424 POKE 764,255
2430 X=USR(1536,ADR(DATA$)):REM TAKE TRACE
2500 REM
2501 REM ---PLOT TRACE IN 4 SEGMENTS,
2502 REM ---256 VALUES IN EACH LINE
2503 REM ---POKE OIRECTLY INTO MEMORY
2504 REM ---WITH THE USR FUNCTION.
2505 REM ---THE AODRESSES USEO ARE
2506 REM ---COMPUTEO IN BASIC  AND
2507 REM ---PASSED AS USR ARGUMENTS.
2508 REM
2510 GRAPHICS 8:POKE 752,1
2520 SETCOLOR 1,6,14
2530 SETCOLOR 2,6,4
2540 COLOR 1
2610 SCREEN=PEEK(106)*256-7373:REM LEFT ADDRESS OF TOP LINE
2620 FOR LINE=0 TO 3:REM EACH 256 BYTE LINE
2630 X=USR(1600,ADR(DATA$)+LINE*256,SCREEN+LINE*1280+640,SCREEN+LINE
*1280,LINEMA SK)
2640 NEXT LINE
2710 FOR LINE=0 TO 3:REM AOO LOGIC LEVEL MARKS ON LEFT
2720 PLOT 20,12+LINE*32
2730 ORAWTO 20,12+LINE*32+16
2740 NEXT LINE
2810 NEXT LOOP
```

MICRO™

NEW FROM D & N MICRO PRODUCTS, INC.



MICRO-80 COMPUTER

Z80A CPU with 4MHz clock and CP/M 2.2 operating system. 64K of low power static RAM. Calendar real time clock. Centronics type parallel printer interface. Serial interface for terminal communications, dip switch baud rates of 150 to 9600. 4″ cooling fan with air intake on back of computer and discharge through ventilation in the bottom. No holes on computer top or side for entry of foreign object. Two 8″ single or double sided floppy disk drives. IBM single density 3740 format for 243K of storage on each drive. Using double density with 1K sectors 608K of storage is available on a single sided drive or 1.2 meg on a double sided drive. Satin finish extruded aluminum with vinyl woodgrain decorative finish. 8 slot backplane for expansion. 48 pin buss is compatible with most OS boards. Uses all standard IBM format CP/M software.

Model 80-1200 2 8″ single sided drives, 1.2 meg of storage	$2995
Model 80-2400 2 8″ double sided drives, 2.4 meg of storage	$3495
Option 001 Serial printer port, dip switch baud rate settings	$ 95

Software available in IBM single density 8″ format.

Microsoft	
Basic-80	$289
Basic Compiler	$329
Fortran-80	$410
Cobol-80	$574
Macro-80	$175
Edit-80	$105
Mu Simp/Mu Math	$224
Mu Lisp-80	$174

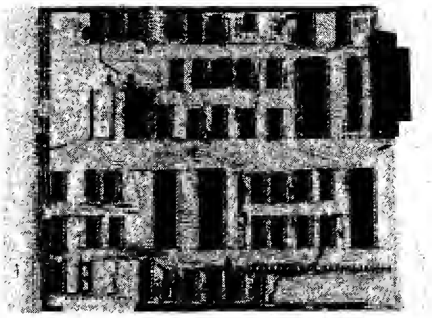
Digital Research	
PL/1-80	$459
Mac	$ 85
Sid	$ 78
Z-Sid	$ 95
C Basic-2	$110
Tex	$ 90
DeSpool	$ 50
Ashton-Tate	
dBase II	$595

Micropro	
Wordstar	$299
Mail-Merge	$109
Spellstar	$175
Super Sort I	$195
Pascal	
Pascal/MT +	$429
Pascal Z	$349
Pascal M	$355

Convert almost any static memory OSI machine to CP/M® with the D & N-80 CPU Board.

Z80A CPU with 4MHz clock. 2716 EPROM with monitor and bootstrap loader. RS-232 serial interface for terminal communications or use as a serial printer interface in a VIDEO system. Disk controller is an Intel 8272 chip to provide single or double density disk format. 243K single density or 608K double density of disk storage on a single sided 8″ drive. A double sided drive provides 1.2 meg of storage. DMA used with disk controller to unload CPU during block transfers from the disk drives. Optional Centronics type parallel printer port complete with 10 ft. cable. Optional Real Time Calendar Clock may be set or read using 'CALL' function in high level languages. Power requirements are only 5 volts at 1.4 amps. Available with WORDSTAR for serial terminal systems.

D & N-80	serial	$695
D & N-80	serial w/Wordstar	$795
D & N-80	video	$695
Option 001	parallel printer and real time calendar clock	$ 80



D & N-80 CPU BOARD

OTHER OSI COMPATIBLE HARDWARE

IO-CA10X Serial Printer Port $125
Compatible with OS-65U and OS-65D software

IO-CA9 Parallel Printer Port $175
Centronics standard parallel printer interface with 10 ft. flat cable

BP-580 8 Slot Backplane $ 47
Assembled 8 slot backplane for OSI 48 pin buss

24MEM-CM9	**$380**	**24MEM-CM9F**	**$530**
16MEM-CM9	**$300**	**16MEM-CM9F**	**$450**
8MEM-CM9	**$210**	**8MEM-CM9F**	**$360**
BMEM-CM9F	**$ 50**	**FL470**	**$180**

24K memory/floppy controller card supports up to 24K of 2114 memory chips and an OSI type floppy disk controller. Available fully assembled and tested with 8, 16, or 24K of memory, with floppy controller (F). Controller supports 2 drives. Needs separated clock and data inputs. Available Bare (BMEM-CM9F) or controller only (FL-470). Ideal way to upgrade cassette based system

C1P-EXP Expansion Interface $ 65
Expansion for C1P 600 or 610 board to the OSI 48 pin buss. Requires one slot in backplane. Use with BP-580 backplane

BIO-1600 Bare IO card $ 50
Supports 8K of memory, 2 16 bit parallel ports may be used as printer interfaces. 5 RS-232 serial ports, with manual and Molex connectors

DSK-SW Disk Switch $ 29
Extends life of drive and media. Shuts off minifloppy spindle motor when system is not accessing the drive. Complete KIT and manual

D & N Micro Products, Inc.
3684 N. Wells St.
Fort Wayne, Ind. 46808
(219) 485-6414

VISA MasterCard

TERMS $2.50 shipping, Foreign orders add 15%.
Indiana residents add 4% sales tax.

Disk Drives and Cables

8″ Shugart SA801 single sided	**$395**
8″ Shugart SA851 double sided	**$585**
FLC-6 6ft. cable from D & N or OSI controller to 8″ disk drive	**$ 69**
5 1/4″ MPI B51 with cable, power supply and cabinet	**$450**
FLC-5 1/4 8 ft. cable for connection to 5 1/4 drive and D & N or OSI controller, with data separator and disk switch	**$ 75**

Okidata Microline Printers

ML 82A Dot Matrix Printer **$534**
120 CPS, 80/120 columns, 9.5″ paper width, friction or pin feed

ML 83A Same as 82A except **$895**
16″ paper width, 132/232 columns with tractor feed

ML 84 Same as 82A except 200 CPS, **$1152**
16″ paper width, 132/232 columns, 2K buffer, dot addressable graphics, with tractor feed

Delayed Reset and Autoboot for the OSI

by Ugo V. Re

The following two circuit modifications to an OSI computer will enhance the capabilities of your system by adding a time delay to the action of the BREAK key (System RESET) and by providing an automatic boot-up of a disk system on power-up, or after a power failure.

How often, while typing, have you hit the BREAK key when you wanted to hit the RETURN key? I've made this mistake many times, and with a disk system it becomes very annoying to re-enter BASIC without booting-up the disk.

One way to correct the problem is to relocate the BREAK key away from the keyboard. Another method is to add a time-delayed action to the BREAK (RESET) key similar to the method used by the C1P series 2 computer. I chose to add a time delay to RESET as this circuit would provide an additional benefit that then can be expanded to provide an automatic boot-up on power-up.

Description of Delayed RESET Circuit

Figure 1a is the Delayed RESET Circuit Diagram. This circuit can be added to any OSI system including the C8P with the RESET key mounted in the computer case.

A resistor placed in series with a capacitor will control the time it takes for a capacitor to fully charge or discharge. Because the capacitor's charge changes slowly it cannot be used to directly control the RESET pin on the 6502. The 6502 requires a rapid change from low to high to RESET the chip. The signal from the capacitor can be used, however, after it has been conditioned to change quickly from a low to high state.

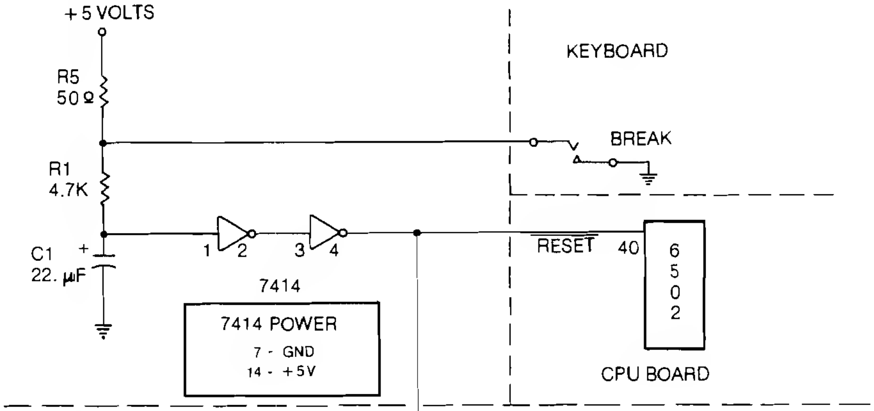
The 7414 IC is a Hex Schmitt Trigger that will produce a rapidly changing output signal from a slowly changing (charging capacitor) input signal. The output goes negative when the input exceeds 1.7 volts, and goes positive when the input drops below 0.9 volts. Two of the six gates are wired in series to invert the signal so that a low (BREAK key depressed) is applied to the RESET pin of the 6502 chip.

Construction of Delayed RESET Circuit

The Delayed RESET circuit can be built on a wire-wrap IC socket as follows:

1. Wire pin 2 to pin 3.
2. Wire pins 5, 9, 11, and 13 to pin 14.
3. Solder the 220 μF capacitor between pin 1 and pin 7 (positive side to pin 1).

Figure 1a: Delayed RESET Circuit



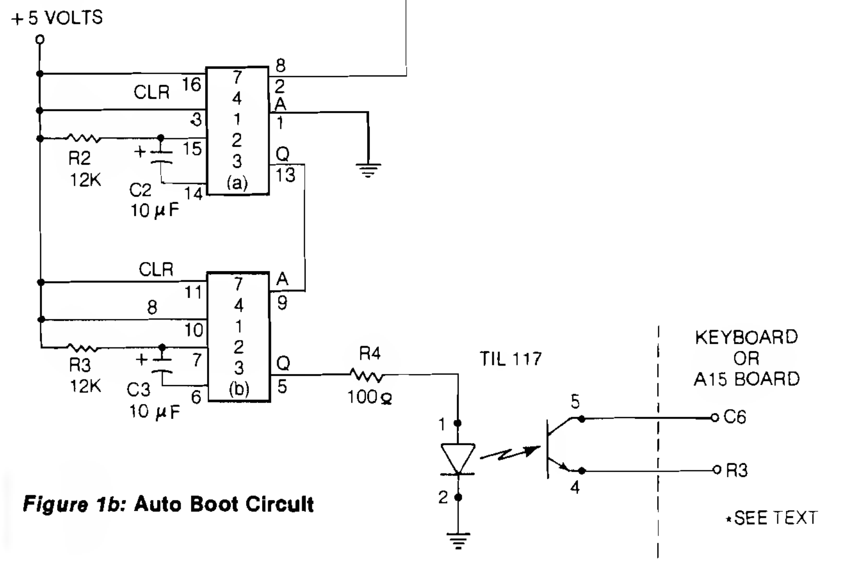


Figure 1b: Auto Boot Circuit

4. Solder a 4.7K resistor to pin 1.
5. Solder a 50Ω resistor to pin 14.
6. Wire pin 14 to 5V and pin 7 to ground.
7. Cut the red lead (trace on the 600 board) that goes from the BREAK key, on the 542 board to pin 40, on the 6502 chip.
8. Solder the loose ends of the 50Ω and 4.7K resistors together and wire to the red lead/trace from the BREAK key.
9. Wire pin 4, 7414 IC to the red lead/trace going to pin 40, of the 6502 chip.
10. Store the IC socket so that it doesn't short out to any other components.

The combination of a 4.7K resistor and a 220 μF capacitor will give a time delay of about two to three seconds. Additional delay can be obtained by increasing the size of the capacitor. Do not increase the size of the resistors as the values shown are the maximum that can be used for reliable operation. With the circuit installed, the BREAK key must be depressed for two to three seconds before the screen will clear.

With the Delayed RESET installed, the computer comes up in RESET at power-up. With a few more components we can use the signal from this circuit to provide an automatic boot-up at power-up.

Description of Auto-Boot Circuit

Figure 1b is the Auto-Boot Circuit Diagram. This circuit provides an additional time delay after RESET goes high before shorting the contacts of the "D" key to boot-up the disk system.

In addition to supplying a signal to the 6502 microprocessor chip, the output of the 7414 IC (Schmitt Trigger) is used to trigger one half of a 74123 IC (Monostable Multivibrator). Its output is then used to trigger the second half of the 74123 in order to produce still another pulse to turn on the TIL117 (Opto-Isolator).

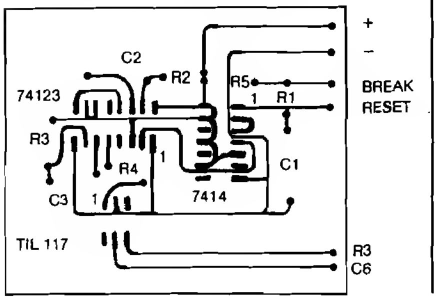
The 74123 can be triggered in two ways. If input A is held low, bringing input B from a low to high state will trigger the 74123. If input B is held high, bringing input A from a high to low state will trigger the 74123. As a result of a trigger, the Q output goes high and the Q output goes low, staying there for a time determined by the resistor and capacitor combination, before returning to their initial states.

At power-up it takes one to two seconds for the capacitor (C1) to charge to approximately two volts before the output of the 7414 will change from a low to high state. This signal is sent to input B1 (first half of the 74123, input A1 grounded), to trigger the 74123. The Q1 output goes high for 100-200 msec before returning to a low state. The Q1 output is connected to input A2 (second half of the 74123, input B2 is wired to 5 volts), therefore, when it goes low it triggers the 74123. The Q2 output goes high and turns on the light-emitting diode (LED) in TIL117. This causes the photo transistor to switch on and short the contacts of the "D" key. After 100-200 msec, Q2 output returns to the low state and the photo transistor returns to an open circuit, with the "D" key no longer shorted. At this time the computer has booted-up BEXEC*.

Construction of Circuit Board

The Delayed RESET and Auto-Boot circuits should be built on a circuit board and mounted at the keyboard. In

Figure 2: Printed Circuit Board



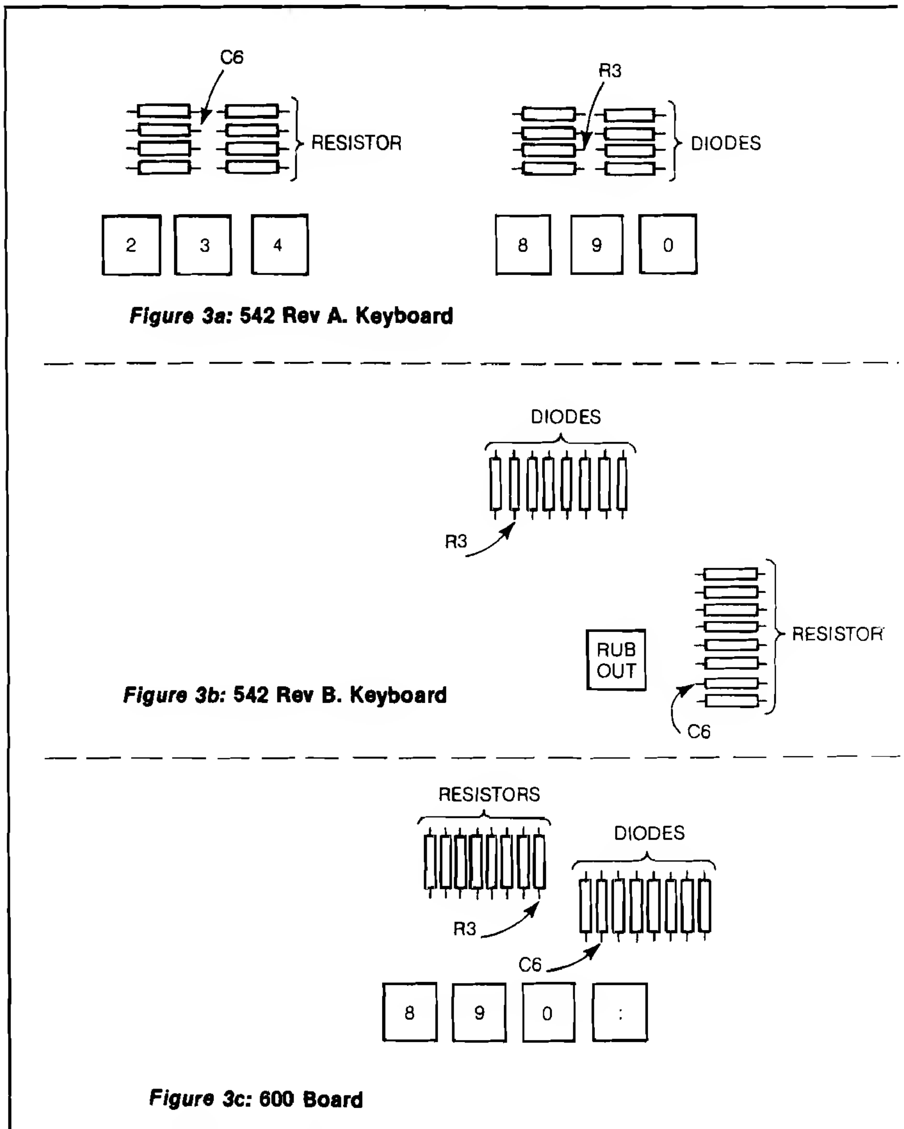


Figure 3a: 542 Rev A. Keyboard

Figure 3b: 542 Rev B. Keyboard

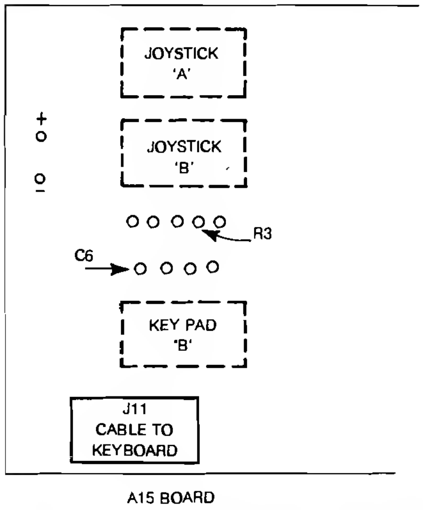
Figure 3c: 600 Board

Figure 3: Location of C6 and R3 Leads

a C8P system it could be placed in the CPU case near the A15 circuit board.

You can use point-to-point wiring with IC sockets mounted on a perforated board, or you can make a printed circuit board (see figure 2 for the layout). Check for the correct orientation of the electrolytic capacitors and integrated circuits before soldering any components to the board.

***Figure 4:* Location of C6 and R3 Leads**



Once the circuit board is completed it must be wired to five volts, ground, the BREAK key, and the contacts of the "D" key. This can be accomplished in one of three ways.

The best method is to wire all leads directly to the contacts on the keyboard. This method is the same for all keyboards. With the keyboard face down, the "D" key contact pads are the fourth group from the right-hand edge and the second group from the bottom edge. The topmost pad (the C6 lead) should have a potential of 4-5 volts and is wired to TIL117, pin 5. The bottom pad is wired to TIL117, pin 4. The BREAK key contacts are also on the right-hand edge of the keyboard. A red and black pair of wires leave the keyboard and go to the CPU board. Power can be picked off at any IC — pin 7 or 8 for ground, and pin 14 or 16 for five volts.

Another method is to wire to a resistor, C6 lead, and diode, R3 lead, located on the keyboard. See figure 3 for the location of these components.

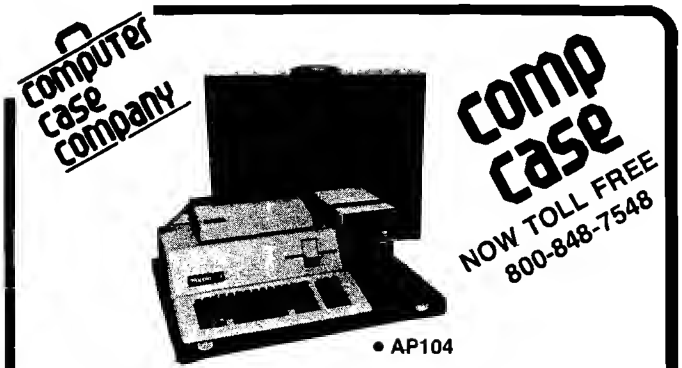
This last method can only be used if you have an A15 circuit board installed in your system. The C6 and R3 leads are extended from the keyboard to the A15 board *via* a 16-conductor cable and appear on pins 2 and 9 of the "A" keypad socket. See figure 4 for the location of socket and wiring points. Power is also available on the A15 board. The BREAK key leads, however, will have to be extended to this board and then routed to the CPU board.

Now whenever you turn on the power, press the BREAK key, or if power returns after a power failure, your computer will boot-up and run BEXEC*. You can modify BEXEC* so that it will run a special program (home control, security, bulletin board, etc.), without any operator intervention. I have my system running continuously and have not yet had a problem.

I would recommend that you do not run your system continuously without some form of motor control for the disks. See the article by N.E. Ingersoll in MICRO, September 1981, page 15. I have this circuit installed in my C4P MF system and highly recommend it to reduce power consumption and disk wear.

The author may be contacted at 167
Sprucewood Dr., Levittown, NY 11756.

MICRO™



The 68000
DREAM MACHINE

Today you can enhance your Pet or Apple II with the power of today's fastest, most powerful microprocessor! Revel in the luxury of sixteen each 32 bit registers! Perform a 32 bit add using **one** instruction, in less than a microsecond! Run graphics over a hundred times faster than Applesoft!

What's that? None of the above interests you? Then **go away** so that we can address those persons who **are** interested!

Now that there is no one here but us performance freaks, let us tell you about DTACK GROUNDED:

Our product is real and available now. It was reviewed in the April PEELINGS II, including a photograph on the front cover. It has also been reviewed in several club newsletters, such as Mini'app'les and the Keystone Apple Core.

In 1981 most people didn't know about the 68000, so we started a newsletter called "DTACK GROUNDED, the Journal of Simple 68000 Systems." We have twelve issues in print, about 180 pages of information about the 68000 and other high performance processors.

About our name: DTACK stands for 'DaTa ACKnowledge'. That's pin 10 on the 68000 and is primarily responsible for the complexity of most 68000 system designs. We tied pin 10 firmly to **ground**, which means that data is **always** acknowledged. So the processor always runs at **full speed** (not true of a lot of expensive 68000-based systems).

Our approach does have some disadvantages. Grounding DTACK prevents hooking some peripherals directly to the 68000. However, we intend that the host computer perform **all** I/O functions (we can transfer 70,000 bytes/sec in either direction between the host and our 68000 board). The operating system is the operating system you **now have** since our board is designed as an attached processor to your personal computer (just the Pet or Apple II for now).

We have 4K boards with 8MHz 68000s. From there we go up to 220K systems with 12.5 MHz (!) 68000s. The latter package is generally purchased for business purposes, of course.

DIGITAL ACOUSTICS
1415 E. McFadden, Ste. F
Santa Ana, CA 92705
(714) 835-4884

ARE YOU BORED?

We designed our 68000 board primarily for the serious individual hacker. We have discovered that there are fewer of those than we anticipated! Did **you** buy one of the first Pets or Apples about four years ago? Remember how much fun you had? Because there wasn't much software back then, many of you wrote your own.

These days every program that **can** be written for your Pet or Apple **has** been written, several times. So, a couple of years ago you started adding hardware accessories. And life became exciting for a few more months. Then, **boredom city** again!

You want to know where all the hackers went? Go look in a mirror. Your problem is that there are no real challenges anymore.

DO WE HAVE A CURE FOR YOU!

When you begin looking into the 68000 you will discover there is little software available right now. That's because the 68000 is less than two years old. There was little software for the 6502 when **it** was less than two years old, you may recall.

We now have new challenges and opportunities. Everything has **not** been done. There have been an enormous number of 68000 based systems announced! Ours is the only **performance oriented** 68000 product which will improve the computer you **already own** (Pets and Apples for now).

You want to have fun? Here is our experience: A simple but elegant 3-D Applesoft graphics demo runs in 53 minutes. We now have the identical package running in 10.3 seconds using a 68000 (with no cheating). That's a speed improvement of over 300 to 1! There is absolutely **no** way that this can be done without a 68000!

What? You **like** being bored? But you want to be bored at 68000 speeds? We have a solution for you: tuck yourself into a time capsule for the next four or five years. When you come out, all of the 68000 software will have been written!

But if you do not have a time capsule, you are going to have a problem very soon now as your computer associates start showing up with machinery that runs circles around your 6502. Aren't bragging rights in your local user group worth a few hundred dollars?

DEALERS: We regret that all sales are factory direct. There is no retail discount. We adopted that policy to make it possible to offer a quality 68000 product at a personally affordable price.

Apple, Applesoft and Apple II are trademarks of Apple Computer Company. Pet is a trademark of Commodore Business Machines.

The 68000 and the Personal Computer

by Laurence Kepple

Your 6502 system is still a gateway to the future! This article discusses some of the hardware and software aspects of what promises to be an exciting market for the 68000 as an add-on to 6502-based systems.

Very Large Scale Integration (VLSI) technology has made it possible for Motorola to turn the CPU architecture of a minicomputer (Digital Equipment Corporation's PDP-11) into a microprocessor. The 68000 is so compact and so powerful that it threatens to erase the distinction between mini and microcomputers. The question now is: do you hold on to the minicomputer concept in the microprocessor world? Do you run two or more terminals on the 68000 as though it were still a PDP-11? Or do you adopt a "one person, one processor" philosophy? The Fortune 32:16 starts at about $5000 and may be able to support as many as 16 terminals. The comparably priced Radio Shack Model 16 can support two terminals. The manufacturers of the SAGE II, which costs under $4000, have deliberately designed their 68000 computer as a single-user system.

Whatever approach is adopted, the rate at which companies are announcing new 68000-based systems makes one thing certain: the 68000 microprocessor has become a chip worth knowing. Fortunately, you do not have to buy an entirely new system to be able to work with the 68000. As is the case with the 6809, add-on boards are becoming available that allow you to use your 6502-based system as a foundation for work with a more advanced processor.

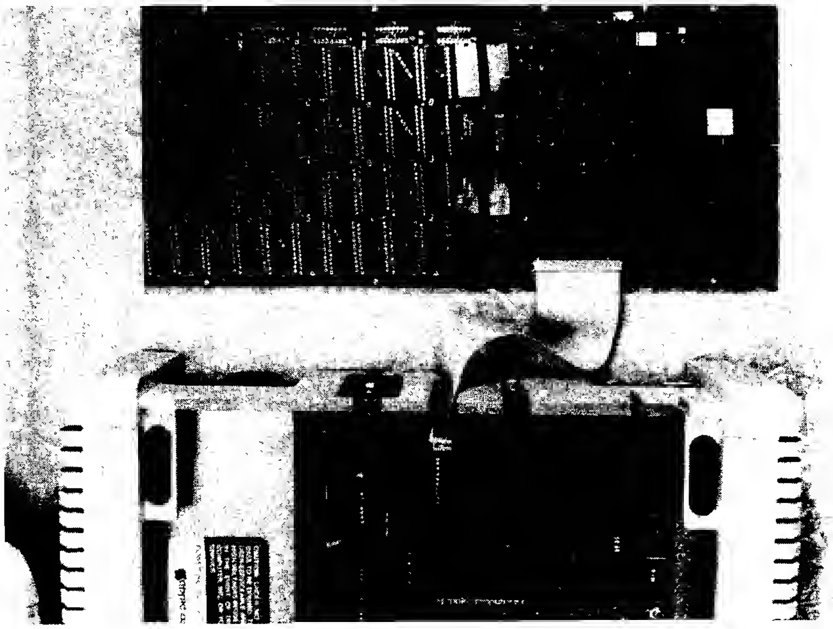
For example, DTACK Grounded (1415 E. McFadden, St. F, Santa Ana, CA 92705) is now shipping peripheral boards for the Apple II and for the PET that allow users to join the 68000 with the 6502 as a co-processor. The cost of these boards ranges from $595 (for an expansion board with 4K of static RAM) to $1123 (for the same board with 92K). (See PEELINGS II, April 1982, for a review of these products.) Company president Hal Hardenbergh, in his newsletter *DTACK Grounded*, has announced that the company will soon market less expensive boards with dynamic instead of static RAM.

The name "DTACK Grounded" explains the company's hardware approach to getting the 68000 into the personal computer market. Pin 10 of the 68000, "DTACK", stands for DaTa ACKnowledged. The 68000 needs this pin to function with the asynchronous data bus for which it was designed by Motorola. By grounding DTACK, the 68000 is made compatible with the synchronous data busses used in simpler personal computer systems.

Using the 68000 as a co-processor with a 6502-based system can produce amazing improvements in execution speed. For example, a 3-D graphics demonstration included with the DTACK Grounded board runs in 30 minutes, 5 seconds in Applesoft; 10 minutes and 40 seconds in Applesoft with the 68000 handling floating point operations; and 45 seconds with the 68000 running a machine-code program and the 6502 used exclusively for I/O. A graphics version of the 68000 floating point package that uses a 16-bit mantissa produced the fastest time of all: 18.9 seconds.

There are other 68000 boards available. Motorola itself recently announced an MC68000 Educational Computer Board. As its name suggests, the purpose of this board is to make it economical (cost: $495.00) for students and designers to get familiar with the 68000. The board includes 32K of RAM and 16K of ROM firmware that provides the user with debug/monitor functions, program entry, assembling, disassembling, and I/O functions. To simplify programming in the educational environment, a primitive assembler/disassembler function allows programs to be entered, displayed, and modified in assembly language. Especially in this latter characteristic, Motorola's board resembles those of

The DTACK Grounded 68000 board (with 12K of static RAM) attached to MICRO's Apple II. The large (!) rectangular shape on the far right of the board is the MC68000.
(Photo by Loren Wright)



DTACK Grounded. Another company, Intellimac (6001 Montrose Rd., Rockville, MD 20852) has announced a 68000 single-board kit, the 68 Magnum, for $745.00. This board offers 128K of dynamic RAM, 16K bytes of firmware ROM/EPROM, and 16K bytes of user EPROM. The system's firmware offers features similar to those of Motorola's ECB. Also like the ECB, the Magnum requires the user to supply a power source and a CRT terminal.

We assume that other companies are going to follow DTACK Grounded's lead and design 68000 boards especially for specific personal computer systems. Micro Technology Unlimited (2806 Hillsborough St., P.O. Box 12106, Raleigh, NC 27605) has announced a 68000 expansion board for its 6502-based system, the MTU-130 microcomputer. Called the Datamover-256, the board provides a 68000 processor and an additional 256K of RAM (price: $999.00). It seems that MTU is deliberately pursuing a 6502/68000 environment, in which users get the best of both worlds: compatibility with a widely supported 8-bit microprocessor, and upgradability to the power of the 68000. At MICRO, we are interested in 68000 peripheral boards aimed at 6502 machines. The significance of such expansion boards goes beyond simply improving the execution time of existing unprotected software. Bringing the 68000 out of the relatively exclusive minicomputer environment into the personal computing world should greatly stimulate the development of software for the 68000. Given the power of the 68000, this would be a significant event. If the 68000 ever achieves anything like the 6502's level of support, the results could be awesome.

Owners of 6502 systems who are considering entry into the 68000 world should be aware that, for some time, they will be strangers in a strange land. The production of software for the 68000 is only beginning. Phase Zero, Ltd. (2509 N. Campbell Ave., Suite 130, Tucson, Arizona 85719) has just started to market the first development software designed specifically for the 6502/68000 environment. Called "The Apple II to MC68000 Cross-Assembler," the program, written in 6502 machine code, accepts 68000 assembly commands and produces 68000 object code (cost: $99.00). The program is intended to run on an Apple II equipped with a DTACK Grounded board. Clearly, i you start to develop 68000 software or your Apple any time soon, you will be one of the first. This is not the case fo those 6502 owners who opt for the 6809 on boards such as those offered by Stellation Two and ESD Labs. The 6809 has a substantial base of excellen software, and two powerful operating systems, OS-9 and FLEX.

However, one crucial event has already taken place that will influence the future of a substantial portion of 68000 development work: UNIX has won wide acceptance as the best operating system for the 68000. One important reason for the UNIX-68000 connection is that UNIX was written for the 68000's minicomputer ancestor, the PDP-11. Created at Bell Labs, UNIX has received hundreds of man-years of additional development at some of America's leading universities. If microcomputer users can gain access to UNIX through the 68000, they will have an operating system of tremendous power. UNIX offers a vast number of utilities and a means of linking them into large constructs *via* "pipes." This makes designing large programs simpler, and encourages the fullest use of the modular approach in solving complex problems. In addition, UNIX is written in, and of course supports, the powerful programming language C.

But the attempt to make UNIX the standard operating system for the 68000 is not without difficulties. In the first place, UNIX itself has yet to be standardized. Because substantial development of UNIX was done at different universities, and because UNIX was designed to be easily modifiable, the operating system assumed a somewhat different form at each site. Variants include Revision 5, Revision 6, Revision 7, and a recently announced System III. Another problem is that, while UNIX provides rich resources in terms of software development tools, it does not offer much application software — again, a situation that reflects the university setting in which UNIX was formed. The absence of an application orientation resulted in some UNIX deficiencies, such as the lack of error recovery for disk I/O. Finally, the language in which UNIX is written, C, while very powerful, is not nearly as portable as COBOL, for instance.

Major efforts are now underway to solve these problems. Microsoft, the company that made BASIC famous, is attempting to achieve the same feat of standardization with UNIX. Microsoft's version of UNIX, called XENIX, supports high-level languages that are well established in the microcomputer market: BASIC, COBOL, FORTRAN, and Pascal. Charles River Data, developer of the $20,000 Universe 68 (the kind of system that Motorola had in mind when it designed the 68000), has also created UNOS to go along with it. In addition to C, UNOS supports FORTRAN, BASIC, and Pascal.

Some software developers have preferred to create independent operating systems that maintain compatibility with UNIX. For example, the Mark Williams Company (1430 West Wrightwood, Chicago, Illinois 60614) has produced the COHERENT system, which it describes as "a totally independent development" that nonetheless is designed to preserve compatibility with Revision 7 of UNIX. Like UNIX, COHERENT was initially developed on the PDP-11. In addition to C, COHERENT supports a BASIC interpreter-compiler, XYBASIC. Mark Williams has just announced a version of COHERENT for the 8088-based IBM Personal Computer. The firm's president, Robert Swartz, noted that existing PC operating systems such as MS-DOS and CP/M are "Model Ts" compared to the "Lear Jet" functionality of a UNIX-like system. A 68000 version of COHERENT is on the way.

Not every 68000 operating system will be UNIX-based or UNIX-compatible. Mindful of the relative lack of applications software that runs under UNIX, some manufacturers will try to use large libraries of applications programs to induce buyers to stay with their non-UNIX but processor-independent operating systems. For example, Microcobol Products, Inc. (MPSI), developer of the BOS/5 and MBOS/5 operating systems, plans to release a 68000 version in the fall of 1982. The BOS/5 operating system is already available on a wide range of processors (Z80, PDP-11, 8086) and supports a large base of business applications programs. Once the BOS/5 system is ported to the 68000, all of this applications software will run without any recompilation or relinking. Another business-oriented, non-UNIX operating system is OASIS 16, the 16-bit version of an 8-bit operating system that supports a substantial quantity of business software.

However, UNIX will almost certainly be the primary operating system used in the development of new applications software for 68000 microcomputers. Some powerful microprocessor development tools, in addition to those already part of UNIX, are being made available for the UNIX environment. The Boston Systems Office (469 Moody St., Waltham, MA 02154) recently announced a delivery date in late June for UNIX-compatible versions of the more than 50 high-speed microprocessor assemblers, symbolic debuggers, and linkers that they originally developed for the PDP-11. Given the close relationship between the 68000 and the PDP-11, this minicomputer is an ideal software development system for 68000 systems running UNIX. BSO's strategy is to support such a development effort on the PDP-11.

The low-cost availability of a chip with the power of the 68000 is good news for anyone who believes in the potential of microcomputers. Perhaps the best news of all is that we do not have to spend $5,000 or more for access to the 68000. Your 6502 system is still a gateway to the future. In time, the development effort at the minicomputer level, in addition to the work done by microcomputer users themselves, will produce microcomputer performance and capacity that we can hardly imagine today. MICRO

(Continued on page 38)

VOLUME 2
now available!

NIBBLE *EXPRESS*

TABLE OF CONTE

NIBBLE EXPRESS VOLUME 2
NOW AVAILABLE!

The 1981 Anthology of the major articles and programs is underway! It contains *up-to-date enhancements and corrections to programs which appeared in Volume 2 of NIBBLE.*

Even if you have all 8 issues of NIBBLE in 1981, you'll want The Express to have updated program listings in one convenient package!

200 solidly packed pages with major programs and articles for your Apple! A MUST for your Library!

NIBBLE EXPRESS will be an invaluable reference for your Apple now and for years to come!

for only $14.95 (plus shipping) you can make Nibble Express a permanent part of your library.

ORDER YOUR COPY NOW!

NIBBLE
P.O. Box 325
Lincoln, MA 01773

Yes! I want NIBBLE EXPRESS Vol. 2 in my library!
Here's my □ Check □ Money order
for $14.95 plus $1.75 postage/handling. (Outside U.S. add $2.75 for postage/handling).

□ Also send me NIBBLE EXPRESS Vol. 1 at $12.95 plus $1.75 postage/handling (outside U.S. add $2.75 for postage/handling).

Master Card & Visa Accepted

Card # ____________________ Expires ______

Telephone ____________________

Name ____________________

Street ____________________

City ____________ State ______ Zip ______

Your check or money order must accompany your order to qualify.
Outside U.S.: Checks must be drawn on a U.S. Bank.
*Apple is a registered trademark of Apple Computer Company.

Chances are, when you bought your first disk drive, it was an Apple. Now that you're ready for a second, take a look at Quentin.

Our Apple*-Mate™ 5¼" Disk Drive is fully software transparent with Apple's DOS 3.3 operating system in full and half track operation.

Add it to your present drive for greater capacity and faster access. Just plug it in and go to work.

And the Apple-Mate has these High Performance advantages:

ON TRACK HEAD SEEK

A precision lead screw positions the head onto the correct track. Time-consuming retries and disk-to-disk copying errors are virtually eliminated.

SIEMENS† DISK DRIVE

The apple-beige unit is built around the highly reliable Siemens system with over 10,000 lifetime hours. Shielded connecting cable also attached.

LONG TERM DEPENDABILITY

MTBF (Mean Time Between Failures)—8,500 power-on hours, and the unit has a one-year warranty.

COUNT ON QUENTIN FOR QUALITY

Quentin Research was building disk systems for the computer industry when Apple was a little bud on the big computer tree. We're known for product reliability and stand behind every system we sell you.

But the best news may be the price—only $335.00 (40 tracks).

A special introductory offer when you order Apple-Mate directly from us.

So when you're ready to boost the juice on your Apple, add-on the Quentin Apple-Mate.

To order: Check, money order, Visa or Mastercard number. Calif. residents add 6% sales tax. Allow one week delivery.

QUENTIN
RESEARCH, INC.

19355 Business Center Drive
Northridge, California 91324
(213) 701-1006

MORE JUICE FOR YOUR APPLE®

Special
Introductory
Price:
$335.00

® Apple is a registered trademark of Apple Computer, Inc.
†Siemens is a trademark of Siemens Corporation.
*Apple-Mate is a trademark of Quentin Research, Inc., which does not manufacture Apple computers.

An MC68000 Overview

by Tom Whiteside and Joe Jelemensky

This article is the first in a two-part series by the authors describing the Motorola MC68000 microprocessor. This part describes the registers, instruction set and addressing modes.

The second part will give some simple programming examples to illustrate programming techniques and special features of the MC68000.

The Motorola MC68000 is rapidly becoming the most popular new-generation microprocessor used in high performance desktop or personal computers. Several manufacturers have announced systems using the MC68000, with costs ranging from $5,000 to $10,000, depending on the amount of memory and size and type of disk storage. These systems pack the power of a minicomputer into a personal computer-size package and price.

The main reason for this favorable performance/price ratio is the advanced micro-programmed architecture of the MC68000. Its 16-bit external data bus and 32-bit internal registers allow high throughput. A 23-bit external address bus, plus upper and lower data strobe, allows direct addressing of 16 megabytes of memory. The powerful instruction set was designed to allow easier implementation of structured high-level languages, such as Pascal and C. Operating systems are also supported by the basic architecture, with features such as:

- supervisor and user states of operation
- privileged supervisor instructions
- separate supervisor stack pointer
- vectored, multilevel, priority interrupt handling
- TRAP vectors to handle a variety of error conditions and operating system calls

Data types include bit, byte, word, long word, and BCD digits. The machine has 17 32-bit registers. Sixteen of these are accessible in user state (8 address registers and 8 data registers). The remaining register is the supervisor stack pointer and is only accessible when the machine is in supervisor state.

The MC68000 was designed to take advantage of the latest technology in VLSI semiconductor manufacturing. Likewise, the recent advances in computer science have also influenced the design of the CPU and the instruction set of the MC68000. The LINK and UNLK instructions make it easy to allocate and deallocate space dynamically on the system stack. This is especially useful when setting up local variables in a procedure. Another instruction, CHK, aids the implementation of high-level languages. CHK allows the checking of array bounds with a single instruction by comparing the value of a register specified with zero and an upper limit.

The instruction set is quite regular, in that most instructions use the same addressing modes and can operate on most data types. All op codes are 16 bits in length. Additional information necessary for specific instructions and addressing modes may require one to four additional words to contain immediate data, displacement offsets, or absolute addresses.

The basic instruction types are: data movement operations; integer arithmetic; logical operations; shift operations; bit manipulation (bit addressable test, set, clear); BCD arithmetic; and program control and system control operations. The complete instruction set is shown in tables 1 and 2.

The addressing modes of the MC68000 are of six basic types: register direct, register indirect, absolute, immediate, program counter relative, and implied. Add to these basic types the ability to do indexing, post-incrementing, and pre-decrementing, and you have 14 specific addressing modes (as shown in table 3).

Registers

The 17 data and address registers free the programmer from the overhead of a complicated register management scheme in the majority of applications. Eight of the registers (D0 through D7) are data registers and may be used for byte, word, and long word operations. The remaining 8 (A0 through A7) are user-accessible address registers. The address registers and the supervisor system stack pointer may be used for word and long word address operations. All of the 17 registers may be used as index registers. All allow post-incrementing and pre-decrementing, giving the programmer the ability to use any of them as a user stack pointer.

The other registers on the MC68000 are a 16-bit status register and a 32-bit program counter. The registers are summarized in the programming model in table 4.

Data Registers

Each of the eight data registers (D0-D7) is 32 bits wide. Data registers are further subdivided into word (16 bits) and byte (8 bits) lengths for data manipulation. Long word operands, of course, occupy the entire register. Word operands occupy the low order 16 bits, and bytes occupy the low order 8 bits. When data registers are used as either a source or destination operand in an instruction, only the appropriate portion of the register is used or changed; the remaining portion is left undisturbed.

***Table 1:* Instruction SET**

Mnemonic	Description
ABCD	Add Decimal with Extend
ADD	Add
AND	Logical And
ASL	Arithmetic Shift Left
ASR	Arithmetic Shift Right
Bcc	Branch Conditional
BCHG	Bit Test and Change
BCLR	Bit Test and Clear
BRA	Branch Always
BSET	Bit Test and Set
BSR	Branch to Subroutine
BTST	Bit Test
CHK	Check Register Against Bounds
CLR	Clear Operand
CMP	Compare
DBcc	Test Condition, Decrement and Branch
DIVS	Signed Divide
DIVU	Unsigned Divide
EOR	Exclusive OR
EXG	Exchange Registers
EXT	Sign Extend
JMP	Jump
JSR	Jump to Subroutine
LEA	Load Effective Address
LINK	Link Stack
LSL	Logical Shift Left
LSR	Logical Shift Right
MOVE	Move
MOVEM	Move Multiple Registers
MOVEP	Move Peripheral Data
MULS	Signed Multiply
MULU	Unsigned Multiply
NBCD	Negate Decimal with Extend
NEG	Negate
NOP	No Operation
NO	Ones Complement
OR	Logical OR
PEA	Push Effective Address
RESET	Reset External Devices
ROL	Rotate Left without Extend
ROR	Rotate Right without Extend
ROXL	Rotate Left with Extend
ROXR	Rotate Right with Extend
RTE	Return from Exception
RTR	Return and Restore
RTS	Return from Subroutine
SBCD	Subtract Decimal with Extend
Scc	Set Conditonal
STOP	Stop
SUB	Subtract
SWAP	Swap Data Register Halves
TAS	Test and Set Operand
TRAP	Trap
TRAPV	Trap on Overflow
TST	Test
UNLK	Unlink

Address Registers

Each address register and the stack pointers are also 32 bits wide. The address registers may also be used for word operands, but, unlike the data registers, may not be used for byte operands. Word operands are contained in the low order 16 bits of an address register. When an address register is used as a source operand, either the entire register or the low order 16 bits will be used, depending on whether it is a long word or word operation. If an address register is specified as a destination operand, the entire register is affected. Long word operations will of course write the 32-bit value into the entire 32-bit register. Word operations will cause sign extension on any other operands before the operation takes place.

The system stack pointers (A7 in user and supervisor state) are used by the machine for interrupt, subroutine call, and trap processing. The two stack pointers are both accessed by using A7 in the operand field of an instruction. The supervisor stack will be accessed if the MC68000 is in supervisor state and the user stack pointer will be used if it is not in supervisor state. The user may also use the system stack to store temporary data by pushing and pulling data *via* the MOVE instruction, or by using the LINK and UNLK instructions described later.

Program Counter

The program counter of the MC68000 is a full 32 bits in length, even though the range of the address space is only 24 bits (23 bits plus upper and lower data strobe) on the current version. Future versions will have 32 bits of address space. The program counter is manipulated by program control-type instructions (branches, subroutine calls, etc.). The program counter is also used as the base address in program counter relative address calculations.

Status Register

The status register on the MC68000 is 16 bits long. It is broken up into an 8-bit system status byte and an 8-bit condition code byte. The system status byte contains an eight-level interrupt mask and additional processor-related status bits. These status bits indicate whether or not the MC68000 is in trace state and/or supervisor state. The condition code byte contains bits for overflow (V), zero result (Z), negative result (N), carry (C), and extend (X). The status word has six unimplemented bits reserved for future extensions to the MC68000 family.

Instruction Set Description

In this section we will discuss the MC68000 instruction set in moderate detail. There are several variations of some instructions, which make them shorter (and/or faster), allow easy manipulation of pointers, and more flexible immediate addressing. The programmer may explicitly choose these variations or allow the Motorola Resident Structured Assembler to select the appropriate form of the instruction. These variations include "quick", "immediate", and "address" instruction types, designated by "Q", "I", and "A" suffixes to the instruction mnemonic.

Data Movement Operations

Unlike the MC6800/6809 architectures, the MC68000 does not have specific load and store register operations containing explicit register reference. Instead, a more powerful MOVE instruction is provided to perform data transfers. The source and destination of data transfer operations are specified independently of the instruction. This allows movement of data between registers, from register to memory, from memory to register, and from memory to memory. Data MOVE instructions allow byte, word, and long word operands. Address MOVE instructions allow only word and long word operands. This insures that only legal address manipulations can be executed.

The MC68000 also includes several special MOVE instructions. The MOVE Multiple register instruction (MOVEM) allows several registers to be moved to or from memory with one instruction. The MOVE Peripheral data instruction (MOVEP) provides a means to transfer word and long word data to or from peripheral data registers, which are located at alternate addresses in memory space. The EXchanGe registers instruction (EXG) allows the swapping of contents between any pair of the accessible registers. Load Effective Address (LEA) and Push Effective Address (PEA) provide means for manipulating address pointers in position-independent programs by loading the effective address of a variable into an address register or pushing it onto a stack. LINK stack (LINK) and UNLinK stack (UNLK) instructions allow the user to easily reserve and return temporary variable space on a stack. The MOVE Quick instruction (MOVEQ) makes it easy to transfer sign-extended 8-bit immediate data to a register with a very fast one-word instruction.

Integer Arithmetic Operations

The four basic arithmetic functions of add, subtract, integer multiply, and integer divide are supported by the MC68000, along with other useful operations such as compare operand, negate operand, test operand, and clear operand.

The ADD and SUB operations are available for both address and data operations. Data operations allow all sizes of operands. Address operations are restricted to legal address size operands (long word and word). The arithmetic CoMPare instruction (CMP) can be used on data, address, and memory compare operations. The NEGate (NEG) and CLeaR (CLR) instruction can be used on all sizes of data operations (byte, word, and long word). The multiply instruction is used to multiply two word operands to form a long word result. MULU is used for unsigned multiplication and MULS for signed multiplication.

In division, a long word dividend is divided by a word divisor to produce a word quotient with a word remainder. Again, instructions are provided for signed and unsigned operations (DIVS and DIVU). A set of extended arithmetic instructions is also available for programming multiprecision and mixed size arithmetic functions. These include ADDX, SUBX, NEGX, and EXT (sign extend). The test instruction (TST) is used to set the condition codes as a result comparing the operand to the value zero.

Another powerful instruction in this category is the Test And Set instruction (TAS). TAS sets the N and Z condition codes based on a compare of the byte operand specified in the instruction with zero. The most significant bit (sign bit) of the byte is then set before the instruction is completed. This is very useful for manipulation of semaphores between processes, since the test and set operation is uninterruptable.

Immediate data of byte, word, and long word length may be added to or subtracted from the data stored at the effective address using the add and subtract immediate instructions (ADDI and SUBI).

A variation of the add and subtract immediate instructions are "quick" add and subtract (ADDQ and SUBQ). These instructions allow the addition or subtraction of immediate data (in the range one to eight) to byte, word, or long word operands. The advantage of these instructions are speed and compactness. They are particularly useful in loops where loop counts must be altered by some small value other than one, but less than nine each time through the loop.

Binary Coded Decimal (BCD) aritl metic operations are also supported b the MC68000. The following multipr cision BCD arithmetic functions ma be performed: add decimal with exten (ABCD), subtract decimal with exten (SBCD), and negate decimal with e tend (NBCD). The BCD arithmeti operations are restricted to byte-lengt operands and can be register-to-registe or memory-to-memory operations.

Logical Operations

The MC68000 provides a full set c logical operation instructions. Logica and (AND), inclusive or (OR), ex clusive or (EOR), and logical comple ment (NOT) are available for all sizes c integer data operands. In addition, a se of immediate instructions, ANDI, ORI and EORI provide logical operations o all operand sizes.

***Table 2:* Instruction Variations**

Instruction Type	Variation	Description
ADD	ADD	Add
	ADDA	Add Address
	ADDQ	Add Quick
	ADDI	Add Immediate
	ADDX	Add with Extend
AND	AND	Logical AND
	ANDI	AND Immediate
	ANDI to CCR	AND Immediate to Condition Code
	ANDI to SR	AND Immediate to Status Register
CMP	CMP	Compare
	CMPA	Compare Address
	CMPM	Compare Memory
	CMPI	Compare Immediate
EOR	EOR	Exclusive OR
	EORI	Exclusive OR Immediate
	EORI to CCR	Exclusive Immediate to Condition Codes
	EORI to SR	Exclusive OR Immediate to Status Register
MOVE	MOVE	Move
	MOVEA	Move Address
	MOVEQ	Move Quick
	MOVE to CCR	Move to Condition Codes
	MOVE to SR	Move to Status Register
	MOVE from SR	Move from Status Register
	MOVE to USP	Move to User Stack Pointer
NEG	NEG	Negate
	NEGX	Negate with Extend
OR	OR	Logical OR
	ORI	OR Immediate
	ORI to CCR	OR Immediate to Condition Codes
	ORI to SR	OR Immediate to Status Register
SUB	SUB	Subtract
	SUBA	Subtract Address
	SUBI	Subtract Immediate
	SUBQ	Subtract Quick
	SUBX	Subtract with Extend

Shift and Rotate Operations

Bidirectional arithmetic and logical shift operations are performed in the MC68000 by executing the ASR, ASL, LSR, and LSL instructions. Rotate operations may be performed with the ROR, ROL, ROXR, and ROXL instructions.

Arithmetic shift operations shift the operand in the direction specified. The last bit shifted out of the operand will be contained in the carry (C) bit and the extend (X) bit of the Condition Code Register (CCR). When the operand is shifted right, the value of the sign bit of the operand (Most Significant Bit) will be preserved., During left shift operations, the Least Significant Bit of the operand will be replaced by zero (0). The overflow (V) bit will indicate if the sign of the operand has changed during the left shift operation.

Logical shift operations shift the operand either right or left as specified by the instruction. The last bit shifted out of the operand in either direction will be contained in the C and X condition code bits. Zeros are shifted into the MSB during right shifts, and into the LSB during left shift operations.

Rotate operations (ROR and ROL) rotate only the bits in the operand and do not include the contents of the extend (X) bit. The operand is rotated in the direction specified by the instruction, and the last bit rotated will also be contained in the carry (C) bit. The contents of the X bit will never be rotated into the operand.

Extended rotate operations are provided by the rotate with extend instructions (ROXR and ROXL), which include the extend (X) bit in the operation. During these operations, the previous contents of the X bit will be shifted into the MSB or LSB, depending on the direction of the rotate operation. Multiple precision shifts and rotates can be performed using a combination of shift and extended rotate operations.

All shift and rotate operations may be performed on memory or registers. When a register is specified as the effective address of the operand, all operand sizes are allowed. A shift count of one to eight bits may be specified in the instruction, or a count of zero to 63 bits may be specified in a data register.

Shifts and rotates on memory operands can only be single bit shifts and are restricted to word length operands.

Bit Manipulation Operations

Several powerful bit manipulation instructions have been implemented in the MC68000. Individual bits may be tested (BTST), tested and set (BSET), tested and cleared (BCLR), or tested and toggled (BCHG). The BTST instruction causes the state of the specified bit in the operand to be reflected by the zero (Z) bit in the condition code register. The BSET and BCLR instructions allow the state of the specified bit to be reflected by the Z bit, and in addition, set or clear the specified bit after the test. The BCHG instruction causes the state of the bit to be reflected in the Z bit, and also changes the state of the specified bit in the operand to the opposite state after the test.

Bit manipulation operations may be performed on register or memory operands. If the effective address is specified as a register, the bit number specification is modulo 32, allowing all bits contained in the register to specified. Memory operands are restricted to byte length and the bit number specification is modulo 8. The bit number is specified as part of the instruction, or in a data register specified in the instruction.

Program Control Operations

Program execution in the MC68000 is controlled by the Program Counter (PC). The PC is updated in the normal processing of an instruction to point to the next instruction to be executed. The PC will either point to the next sequential instruction in memory, or to another instruction sequence elsewhere in memory as a result of a PC modifying instruction.

The program counter is modified by the execution of an instruction in one of two ways. In the first way, the PC is modified by adding a displacement, specified in the instruction, to the current PC to get the address of the next instruction to be executed. The displacement (word or byte) specified in relative branch instructions is a two's complement integer value, which counts the relative distance in bytes.

***Table 3:* Addressing Modes**

Mode	Generation
Register Direct Addressing	
Data Register Direct	EA = Dn
Address Register Direct	EA = An
Absolute Data Addressing	
Absolute Short	EA = (Next Word)
Absolute Long	EA = (Next Two Words)
Program Counter Relative Addressing	
Relative with Offset	EA = (PC) + d_{16}
Relative with Index and Offset	EA = (PC) + (Xn) + d_8
Register Indirect Addressing	
Register Indirect	EA = (An)
Postincrement Register Indirect	EA = (An), An ← An + N
Predecrement Register Indirect	An ← An - N, EA = (An)
Register Indirect with Offset	EA = (An) + d_{16}
Indexed Register Indirect with Offset	EA = (An) + (Xn) + d_8
Immediate Data Addressing	
Immediate	DATA = Next Word(s)
Quick Immediate	Inherent Data
Implied Addressing	
Implied Register	EA = SR, USP, SP, PC

Notes:

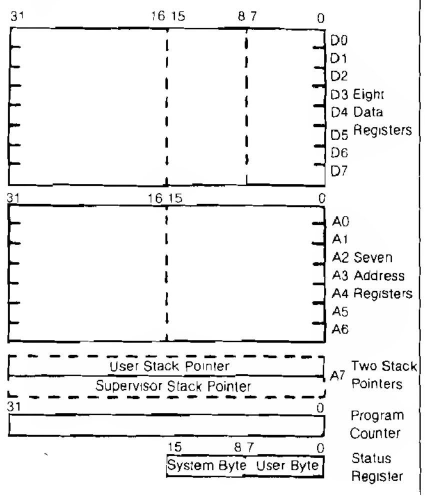
EA	= Effective Address
An	= Address Register
Dn	= Data Register
Xn	= Address or Data Register used as Index Register
SR	= Status Register
PC	= Program Counter
( )	= Contents of
d_8	= 8-bit Offset (displacement)
d_{16}	= 16-bit Offset (displacement)
N	= 1 for Byte, 2 for Words and 4 for Long Words
←	= Replaces

The value of the PC is the address of the current instruction, plus two. The displacement is calculated by the assembler when the program is assembled, or by the linker when building a load module. By adding displacements to the current PC, the program will be position independent, since all PC changes are relative and control is always resumed at the same location in the program instead of an absolute memory address. This allows programs to execute anywhere in memory instead of from a specific area.

In the second way, the PC may be modified by instructions that load an effective address value into the PC when the instruction is executed. In this case, the value of the PC will be overwritten by the new address specified in the instruction. When these instructions are executed, control will be transferred to the memory address specified regardless of where the program is loaded.

Program control operations are accomplished with a collection of branch instructions, subroutine branches, and return instructions. Branch instructions can be conditional or unconditional. The conditional branches allow testing various condition bits in the condition code register individually or in Boolean combination. If the condition specified in the branch instruction is met, the branch will be taken. If not, the next sequential instruction will be executed. The conditional branch instruction (Bcc) tests the condition specified in the instruction. Then it either branches or falls through to the next instruction, based on whether the condition is met or not. The test condition, decrement and branch instruction (DBcc), tests the specified condition and falls through if it is true. If the condition is not true, the contents of a specified register is decremented and the result compared to −1. If the value is not −1, the branch will be taken. Otherwise control will resume with the instruction following the DBcc. All conditional branches are relative to the current PC. Table 6 lists the conditions that can be tested by the conditional branch instruction.

Table 4: Programmer's Model for the MC68000



Unconditional branch instructions always force the destination address of the branch to be loaded into the program counter. These branches may be relative to the current PC or absolute addresses. The BRAnch unconditional instruction (BRA) will always cause the PC to be replaced by the value of the current PC, plus the relative displacement specified in the instruction. The JuMP instruction (JMP) causes the PC to be loaded with the effective address specified in the JuMP instruction.

There are also two subroutine call instructions. These instructions first save the value of the current PC (address of next sequential instruction to be executed) on the system stack (pointed to by address register A7), and then cause the PC to be loaded with the address of the subroutine to be executed. The Branch to SubRoutine (BSR) instruction causes the relative offset specified in the instruction to be added to the PC. The Jump to SubRoutine instruction (JSR) loads the PC with the effective address of the subroutine specified by the effective address specified by the JSR instruction.

There are two instructions that return program control to the address stored on the system stack. ReTurn from Subroutine (RTS) pulls the new program counter value from the stack and program execution resumes at that address. The ReTurn and Restore instruction (RTR) additionally restores the condition code register from the stack when returning from interrupt processing.

Bcc and BSR can have displacement values of byte and word lengths. DBcc always has a word length displacement value. The effective address used by the JMP and JSR instructions is specified by the addressing mode of the instruction.

Another instruction that uses the condition codes is the Set according to condition (Scc). The Scc instruction tests the specified condition (see table 6) and sets the byte specified by the effective address to all ones if the condition is true. Otherwise the byte is cleared.

System Control Operations

Several instructions offer easy implementation of system control functions. These instructions allow logical operations on the status and condition code bits, trap and exception processing, special range checking, special program flow control operations, and even a special peripheral reset operation.

Many of these can be used only if the MC68000 is in the supervisor mode (S bit in the system byte of the Status Register set) to support the operating system environment. These privileged instructions include those that change the state of bits in the system byte of the status register (ANDI, EORI, ORI, and MOVE EA to SR), change the value of the User Stack Pointer (MOVE USP), ReTurn from Exception trap (RTE), stop program execution (STOP), and cause the RESET pin to become an output and be asserted (RESET).

Instructions that directly change the contents of the Condition Code Register portion of the status register (ANDI, EORI, ORI, and MOVE EA to CCR) and store the status register (MOVE SR to EA) may be executed in user mode also.

The user program may also execute intructions that might generate a trap. These are: the range checking instruction (CHK); the specific trap generation to one of 16 trap vectors (TRAP); and causing a trap on overflow bit set in the condition code register (TRAPV).

Addressing Modes

The MC68000's 14 addressing modes can be grouped into three broad classes. These are: 1. Register District, 2. Memory, and 3. Inherent addressing.

Register Direct Addressing Modes

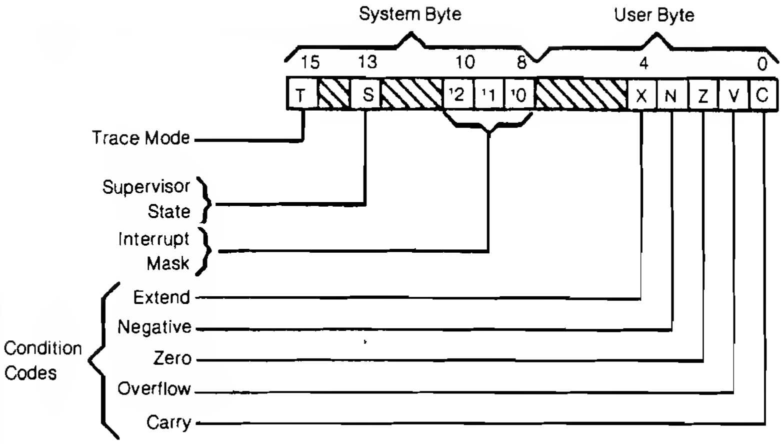
In Register Direct addressing, the effective address is one of the 16 registers. Register addressing can be divided into these two categories: 1. Data Register Direct — Dn; 2. Address Register Direct — An. The notation for Data Register Direct addressing is "Dn" (e.g., D0 refers to data register 0). Likewise, Address Register Direct addressing notation is "An".

Memory Addressing Modes

Unlike Register Direct addressing, Memory addressing modes provide the address of an operand that is in memory. A list of Memory addressing modes for the MC68000 follows. Beneath each of the Memory addressing modes is an example of its assembler syntax. All indirect references are placed in parentheses.

1. Indirect Addressing
   a. Address Register Indirect
      (An)
   b. Address Register Indirect with Postincrement
      (An) +
   c. Address Register Indirect with Predecrement
      − (An)
   d. Address Register Indirect with Displacement
      disp(An)
   e. Address Register Indirect with Index
      disp(An + Im. W) where Im is Am or Dm
      disp(An + Im. L) where Im is Am or Dm
2. Absolute Addressing
   a. Absolute Short
      inst.W ADDRESS

***Table 5:* MC68000 Status Register**



   b. Absolute Long
      inst.L ADDRESS
3. Program Counter Relative Addressing
   a. Program Counter with Displacement
      disp(PC)
   b. Program Counter With Index
      disp(PC + In) where In is An or Dn
4. Immediate Addressing
   a. Immediate
      #xxx
   b. Quick Immediate
      #xxx

(continued)

MICRObits

(Continued from page 29)

AIM 65 — Real Time Clock

Provides hour, minute, second, day of week, day, month, year. Twelve- or 24-hour format. Pin compatible with AIM expansion connector (also SYM, KIM). Four switch selectable interrupts. Nicad battery backup. Industrial quality board 4.5 × 6. All ICs socketed. Single 5V supply. Twenty-two-page manual. All software included. Bare board $29. Complete A&T $93, includes batteries. Add $4 shipping and handling. CA residents add 6%.

Data Design Group
7100 Mimosa Drive
Carlsbad, CA 92008
(714) 265-6940

The Wrath of Khan

Can you defeat a Superman? Command Enterprise in a tactical battle against the Reliant. C1-P controls Reliant's maneuvers and weapons *via* Artificial Intelligence logic routines. Features full status display, sensors, photon torpedoes, phasers, deflectors, and more. Cassette 8K - $14.95 ppd.

Cygnus Software
Suite 432
Ft. Lauderdale, FL 33311

Perseus Programming for OSI

Boxed In — An adventure in which you must escape to become a member of the CIA. *Rookie Mission* — Your first mission as an agent and you must find the traitor! *Suicide Mission* — An exciting graphics game that pits you against an alien who has captured your crew, and you must retrieve them against over-whelming odds. Tough! All programs run in 8K and on a C1P, C2P, C2-4P, or C8P with a cassette-based system. Send $9.95 for each or $22.95 for all.

Perseus Programming
9311 Avery Rd.
Broadview Heights, OH 44147

OSI States and Capitals Drill

Let your kids use the computer and learn at the same time! You won't believe this program is running on your 8K C1P! Graphics, screen clears, no scrolling, and it even gives you a grade. Get a correct answer and watch the flag wave. Cassette - $10.95 plus $1.50 for postage. Math drill games also available.

Tripod Productions
Rt. 11, Box 71
Bowling Green, KY 42101

(Continued on page 98)

***Table 6:* Condition Codes Allowed for the DBcc, Scc, and Bcc Instructions.**

CC	Carry Clear	LS	Lower or Same
CS	Carry Set	LT	Less Than
EQ	Equal	MI	Minus
F	False *	NE	Not Equal
GE	Greater or Equal	PL	Plus
GT	Greater Than	T	True
HI	Higher	VC	Overflow Clear
LE	Less or Equal	VS	Overflow Set

* The false condition does not apply to Bcc.

The various forms of Address Register Indirect addressing provide great flexibility. In the simplest of these, the effective address is the contents of the specified address register. In the Post-increment mode, the effective address is still the specified address register but the address register will be incremented *after* the operation. The Pre-decrement mode is the same except the address register is decremented *before* the operation. The Pre-decrement and Post-increment modes automatically adjust the number of words to decrement or increment based on the size of the operation. For the Address Register Indirect with Displacement mode, the effective address is formed by summing the contents of the specified address register with a sign-extended 16-bit offset. The Address Register Indirect with Index effective address is the contents of the specified address register, summed with the sign extended 8-bit displacement and the specified index register. The index register can be any of the address or data registers and can be specified as either a long word or a word (which will be sign extended). The execution speed of the Address Register with Index is not affected by the size of the index.

In the Absolute Addressing modes, the effective address is specified in the word or words following the opcode. In the case of Absolute Short addressing, the word is sign-extended to form the effective address.

The Program Counter addressing modes use the contents of the program counter to form an effective address. These modes are important for writing position-independent code. The Program Counter with Displacement mode sums the contents of the program counter with a 16-bit sign-extended displacement just like Address Register Indirect with Displacement. The Program Counter with Index mode is identical to Address Register Indirect with Index, except that the program counter is used instead of an address register.

For normal immediate addressing, the effective address is the word or long word immediately following the opcode, depending on the size of the operation. Single bytes are stored as a word (the upper byte is zeros). In Quick Immediate addressing, a special case of immediate addressing, the immediate data (a number from 1 to 8) is contained in a 3-bit field in the opcode. Quick Immediate addressing is available for ADD and SUB instructions where constants are used frequently. Immediate addressing limits the destination addressing mode to register direct. However, many special immediate instruction variants exist, such as ADDI, for using memory addressing for the destination address.

Implicit Addressing

Many instructions make implicit references to the program counter, the supervisor stack pointer, the user stack pointer, or the status register. For example, the RTS (return from subroutine) implies use of the program counter and the stack pointer.

Using the MC68000

In the second part, we hope to provide a better feel for using the MC68000 through a series of simple examples, In addition to showing assembler syntax, these examples will illustrate instructions such as LINK, UNLK, and CHK which may be unfamiliar.

Joe Jelemensky and Tom Whiteside are engineers in the Motorola MOS Microprocessor Design Group. They can be contacted at Motorola, Inc., Mail Drop M2880, 3501 Ed Bluestein Blvd., Austin, Texas 78721.

GET FREE SOFTWARE FOR YOUR COMPUTER!

HOW? JUST ORDER ANY OF THE ITEMS BELOW, AND SELECT YOUR FREE SOFTWARE FROM THE BONUS SOFTWARE SECTION, USING THE FOLLOWING RULE: FOR THE FIRST $100.00 WORTH OF MERCHANDISE ORDERED TAKE 1 ITEM; FOR THE NEXT $200.00 WORTH OF MERCHANDISE ORDERED TAKE ANOTHER ITEM; FOR THE NEXT $300.00 TAKE A THIRD ITEM, ETC. ALL AT NO COST.

HARDWARE by APPLE COMPUTER

APPLE II + . 48K	1199	PASCAL	150
FLOPPY OR + CNTRLR	545	FLOPPY DRIVE	465
APPLE III. 128K	2999	PILOT	125

We carry the rest of the APPLE line at low. low prices! CALL!

OTHER HARDWARE for APPLE

0.C. HAYES:

Micromodem II	285	Smartmodem	225

MICROSOFT:

Z80 Softcard	269	16K Ramcard	139

MOUNTAIN COMPUTER.

Expansion Chassis	559	Music System	339
A/0 + 0/A Card	299	CPS Multi-function	169
X/10 Control Card	169	Super Talker	169

CALL FOR MORE PRICES! WE CARRY FULL LINE!

CALIFORNIA COMPUTER SYSTEMS:

Centronics Par Int	115	A/0 Converter	105
Async Serial Int	135	Calendar/Clock	101

CALL FOR MORE PRICES! WE CARRY FULL LINE!

VIOEX:

80 Col Bd & Softswitch	235	Enhancer II	125
Enhancer I	105	Softswitch	29

MORE OTHER HARDWARE FOR APPLE:

SSM AIO-II	195	Keybd Co Num Keypad	129
SSM Serial AS10	115	Sunshine Joystick	39
SSM Par AP10	99	Game Paddles	29
Novation Apple Cat	319	Shadow/Vet	885
Versawriter Tablet	249	SUP'R' MOD	29
Prac Periph Microbuff (32K)	249	Prac Periph Microbuff (16K)	210

OTHER SOFTWARE for APPLE:

PERSONAL SOFTWARE/VISICORP:

Visicalc 3 3	195	Visifiles	199

CALL FOR MORE PRICES! WE CARRY FULL LINE!

MICROSOFT:

APPLE Fortran (Z80)	129	APPLE Cobol (Z80)	499
TASC Basic Compiler	139	MBASIC Compiler (Z80)	299
TIME Manager	125	MuMath	199
ALDS	99	M/SORT	149

MICRO-PRO:

Wordstar	225	Mail-Merge	99
Spellstar	149	Data-Star	199
Super-Sort	149	Calc-Star	149

PEACHTREE. BIZ Packages, all	199
CPA BIZ Packages, all	195

MORE OTHER SOFTWARE for APPLE:

OB Master	179	OB Master for CORVUS	399
Data Factory 5 0	239	PFS	85
ASCII Express	55	Dakin 5 Deprec Planner	299
Sorcim Super Calc	189	Dakin 5 BIZ Bookkeeper	299
Howard Tax Prep	115	Broderbund Payroll	325
Howard Real Estate Anal	129	BPI Accounting Pkgs/ea	325

Synergistic "Data Reporter" Manages Plots & Edits Data!	189

WORO-PROCESSORS & SPELLERS for APPLE:

Wordstar CP/M	225	EZWriter Prof Sys	215
Magic Wand CP/M	299	MUSE Super Text 80	150
Executive Secretary	199	Wordpower	50
Letter Perfect	125	Hebrew II	55
Magic Window	85	Screenwriter II	110
Spellguard	219	Spellstar CP/M	149

Word Handler (Gives 80-Col & Lower-Case with no board!)	199

PRINTERS

EPSON:

MX80	449	MX80 F/T	549
MX100 w/Graftrax	729	MX70 w/Graftrax	285
APPLE Intfce/Cbl	85	GRAFTRAX	60
GRAPPLER Intfc	149	2K Buffer Serial Card	135
MX80 Ribbon	15	MX100 Ribbon	24

C.ITOH:

F-10 Daisy Wheel (Par)	1495	F-10 Daisy Wheel (Ser)	1495
Pro-Writer (Par/Ser)	599	Pro-Writer (Par)	499
F-10 Tractor Option	225	Printer Interfaces	CALL

NEC:			
PC-8023A	495	NEC 7710 Daisy	2345
OUME:		**OIABLO:**	
SPRINT 9/45	1995	630 R/O	2099

OKIOATA:

Microline 82A	495	Microline 80	375
Microline Tractor	59	Okigraph I	79
Microline 83A	799	Microline 84 (Par)	1099

IDS:

560 with graphics	1095	Prism-Print Software	49
Prism 80 (Basic)	899	Prism 132 (Basic)	1050
Auto Sheet Feed	125	Prism Color	325
Sprint Mode (200 cps)	125	Dot Plot Graphics	85

ADD-ON MEMORY CARDS & DISK DRIVES FOR APPLE

MEMORY:

Microsoft 16K Ramcard	139	Saturn 32K Card	199
Legend 128K Ramcard	649	Saturn 64K Card	369
Legend 64K Ramcard	299	Saturn 128K Card	525
SVA 256K APL-Cache	1045	Prometheus 128K	439
AXLON 320K Ram Oisk	1149	16K of 4116. 200 NS Mem	25

APPLE-COMPATIBLE FLOPPIES by MICRO-SCI:

With Controller		No Controller:	
A35 Exact Replacement	460	A35 Exact Replacement	415
A40 40-Track	489	A40 40-Track	399
A70 70-Track	599	A70 70-Track	499

8" FLOPPY DISK SYSTEMS:

Vista Oual SSOD	1299	Vista Dual OSOD	1599
SVA AMS8000 Dual SSDD	1945	SVA AMS8000 Oual DSOD	2595
SVA ZVX4 Quad Cntrllr	495	SVA Disk 2 + 2 Cntrllr	359

CORVUS HARO DISKS:

6 MB Hard Oisk	2249	Apple Interface	175
11 MB Hard Disk	3945	Other Computer Intfce	CALL
20 MB Hard Oisk	4769	Mirror Back-Up	675

MONITORS, PLOTTERS & PERIPHERALS

MONITORS:

Zenith 12" Green	125	Zenith 13" Color	359
Amdek 12" Green	135	Amdek 13" Color	359
BMC 12" Green	119	BMC 12" Color	349
Electrohome RGB Intfce	275	Electrohome RGB 13" Clr	725

PLOTTERS:

Watanabe 1-Pen	1150	Watanabe 6-Pen	1400
Slrobe Plotter 1-Pen	699	Strobel Apple Intfce	99
Houston Inst OMP-3	929	Houston Inst OMP-4	1185

OTHER PERIPHERALS:

Scott Shadow/VET	885	Street Echo II Synth	189
Votrax Type-N'-Talk	325	Computer Sta Oithertizer	269
ALF 9-Voice Music Bd	155	Comp Sta Video Camera	375
ALF 3-Voice Music Bd	199	Comp Stal Both above	599

COMPUTER SYSTEMS

ATARI

ATARI 800 (16K)	629	ATARI 400 (16K)	335
810 Disk Orive	449	825 Printer	599
16K Ram Memory	89	850 Interface	159
Microsoft Basic	69	830 Modem	149
INTEC 32K Ram Memory	129	Telelink Cartridge	25
ATARI 800 (48K)	739	AXLON 128K Ram Oisk	550
		ATARI 400 (48K)	435

OSBORNE

Osborne 1	1695	Printer Cable	55

XEROX

820-1 System w/5" Or.	2450	820-2 System w/8" Or.	2950
CP/M Op Sys	159	Wordstar	419
Super Calc	199	OIABLO 630 Printer	2099

Systems-Plus BIZ SOFTWARE, PER MOOULE	CALL!

NEC

Full Line at Low. Low Prices! CALL!

COMMOOORE VIC

VIC 20 Computer	259	VIC 1515 Printer	335
VIC 1540 Disk Drive	499	VIC 1211A Supr-Expndr	59
VIC 1530 Datasette	69	VIC 1011A RS-232 Port	45

GENERAL CP/M SOFTWARE

MICROSOFT:

Basic 80	275	Edit 80	139
Basic Compiler	299	Mu Math/Mu Simp	199
Fortran 80	339	Mu Lisp/Mu Star	159
Cobol 80	499	M-Sort	128
Macro 80	139	Z-80 Softcard/Apple	269

MICROPRO:

Wordstar	275	Calcstar	199
Mailmerge	89	Supersort	165
Spellstar	165	Custom Notes	275
Datastar	239		

ASHTON-TATE:

dBASE II	475	dBASE II Guide	29

SORCIM:

Super Calc	189		

FOX-GELLER:

Quickscreen	129	dUTIL	69
Ouickcode (Writes programs for dBASE II)			199

ISA:

Spellguard	219	SP/LAW	99

PEACHTREE:

Gen Ledger	399/40	Inventory	399/40
Acct Rec	399/40	Magicalc	269/25
Acct Pay	399/40	Sales Invoicing	399/40
Payroll	399/40		

SYSTEMS PLUS

All Modules	415/EA

S-100 BOARDS

HAYES S-100 MODEM	325	HAYES SMARTMODEM	219
HAYES CHRONOGRAPH	189	CCS 64K RAM BOARD	525

FLOPPY DISKS

Elephant 5 25" Soft SSSD/bx	25	Elephant 8" Soft SSSD/6x	29
3M 5" Soft SSSO/bx	27	3M 8" Soft SSSD/bx	32
Maxell 5" Soft SSSD/bx	31	Maxell 8" Soft SSSO/bx	35
DYSAN 5" Soft SSSO/bx	37	DYSAN 8" Sof SSSD/bx	49

BONUS SOFTWARE SECTION!

Let us acquaint you with MESSAGE MAKING SOFTWARE Just place the disk in the APPLE, enter the text, and colorful dynamic messages appear on the screens of TV sets connected to the computer Use the software to broadcast messages on TV screens in schools, hospitals. factories, store window, exhibit booths, etc The following program is our latest release

SUPER MESSAGE Creates messages in full-page "chunks' Each message allows statements of mixed typestyles, typesizes and colors, in mixed upper and lower case Styles range from regular APPLE characters, up to double-size, double-width characters with a heavy, bold font Six colors may be used for each different typestyle Vertical and horizontal centering are available, and word-wrap is automatic. Users can chain pages together to make multi-page messages Pages can be advanced manually or automatically. Multi-page messages can be stored to disc or recalled instantly
REOUIRES 48K & ROM APPLESOFT $50

APPLE PLOTS YOUR DATA & KEEPS YOUR RECORDS TOO! APPLE DATA GRAPH 2.1 Plots up to 3 superimposed curves on the Hi-res Screen both X & Y axes dimensioned Each curve consists of up to 120 pieces of data. Graphs can be stored to disc and recalled immediately for updating Up to 100 graphs can be stored on the same disc Great for Stock-market Charting, Business Management, and Classroom instruction!
REOUIRES 48K & ROM APPLESOFT $35

APPLE RECORD MANAGER. Allows complex files to be brought into memory so that record searches and manipulations are instantaneous Records within any file can contain up to 20 fields, with user-defined headings Information can be string or numeric Users can browse thru files using page-forward, page-backward or random-search commands Records can easily be searched, altered or sorted at will Files can be stored on the same drive as the master program, or on another, if a second drive is available Records of files can be printed, if desired Additional modules coming are a STATISTICS INTERFACE, CHECKBOOK, MAILING LIST & DATA-ENTRY.
REOUIRES 48K & ROM APPLESOFT . $40

APPLE LITERATURE DATABASE Allows rapid retrieval (via keywords) of references from total APPLE literature thru 1980, on 5.25" disk. Each entry in the data base consists of the article, author-name, periodical-name. date of issue, & page nos The database is intended to support large magazine files which would require lengthy manual searching to recover information. Annual updates will be available
REOUIRES 48K & ROM APPLESOFT . $60

WORDPOWER Is a simple, powerful, low cost, line oriented word processor program It offers a fast machine language FIND & REPLACE. Text can be listed to screen or printer with or without line-numbers. Lower-case adaptors are supported You can merge files, move groups of lines, and easily add, change, or delete lines WORDPOWER can be used to create and maintain EXEC files It can also be used as a rapid, unstructured, information-storage and retrieval system via its rapid search capabilities
REQUIRES 48K & ROM APPLESOFT $50

LABELMAKER. Allows users to quickly create address labels A given label may be generated in any quantity from 1 to 32767. Space is allowed on labels for a personal and company name, but the space is automatically closed up if only a personal name is entered Space is also allowed for foreign countries. The program can also generate labels for price-tags, part numbers and mail-messages such as "RUSH", "FRAGILE", etc A self-incrementing feature allows theatre-tickets to be produced, with a date, and numbers running from a000 to z999 An editor is provided for editing labels prior to printing All labels may be saved to disk for instant recall
REOUIRES 48K & ROM APPLESOFT $35

Above software for APPLE DOS 3.2/3.3 only. Call for BONUSES for other systems.

TO ORDER: Use phone or mail. We accept VISA, MASTERCARD, COD's, personal checks & money orders. Add 4% for credit card. Customer pays handling on COD orders. Foreign orders must be in American Dollars & include 10% for handling. Connecticut residents add 7.5% sales tax. Prices subject to change without notice.

Not responsible for typographical errors Prices subject to change without notice

CONN. INFO. SYSTEMS CO.
(203) 579-0472
218 Huntington Road, Bridgeport, CT 06608

IMAGE PRINTER SERIES

Sensible Software is proud to introduce our new series of high resolution screen dumps. IMAGE PRINTERS provide a simple way to transfer high resolution graphic images onto paper. Each program in the series has unique features that give you full control of the printing. Some of the included options are:

- Full control over the area of the HIRES screen to be printed. You graphically pick the area for the utmost ease and accuracy.
- One-step printout of the picture with the ability to pause or abort the printing at any time.
- Menu-driven. All options are invoked with single keystrokes. IMAGE PRINTERS are extremely easy to use.
- Multiple image sizes, 6 different sizes for letter quality printers, 4 sizes for other printers.
- Creation of an inverse (negative) image for reverse printing.
- The ability to save the compressed and inverse images to disk.
- One time configuring for your printer and interface card. Why answer all those questions about your printer each time you want to print a picture?
- The images may be printed anywhere on the page.
- IMAGE PRINTERS support most popular interface cards, such as cards from Apple, California Computer Systems, Epson, and Mountain Computer. (The SSM AIO Serial Card and user-written 'driver' routines may be used with the letter quality printers.)

There are three separate versions of IMAGE PRINTERS, each one tailored to take full advantage of a different printer.

IMAGE PRINTER—LETTER QUALITY. For all popular letter quality printers (Diablo, NEC, Qume, etc.)

IMAGE PRINTER—EPSON. For the popular Epson MX-70, MX-80 and MX-100.

IMAGE PRINTER—NEC PC-8023A. For the NEC dot-matrix printer.

All versions are available for $40.00 ea.

Please specify version desired.

THE BUG

A sophisticated, yet easy to use diagnostic aid for getting "the bugs" out of your assembly language programs.

If you are a novice just getting started with assembly-language programming, you will find The BUG helpful in developing your understanding of how the Apple's 6502 internal processor operates. The many display options of The BUG will permit you to try out your assembly-language programs at the speed that is most comfortable for you. The BUG will also make it easy for you to see the effect of your program on the Apple as it executes

If you are a professional programmer, you will also find that The BUG can improve your efficiency by reducing the time you spend identifying and solving complex, assembly-language programming errors You will particularly appreciate the fact that The BUG offers the easiest to use and most extensive breakpointing capability of any "debugger" available for the Apple. Up to 13 different breakpoints can be specified to halt program execution when either 1) a particular program location is reached, 2) one of the 6502 registers reaches a specified value or 3) one of the bits in the 6502 status register reaches a specified value

Another key feature of The BUG that serious programmers will appreciate is the ability to AUTOMATICALLY run lower-level subroutines at FULL SPEED You no longer have to keep debugging the portions of your program that you already have working

This is not the least expensive "debugger" program for the Apple, but we challenge you to find more capability for less money!

The BUG is supplied with a 40+ page user guide and is designed for use with DOS 3.3 on either the Apple II or Apple II Plus computer only **$50.00**

BUILD USING

provides an easy to use print-using routine plus similar functions for strings. Creating charts, reports and general screen formatting becomes a simple task BUILD USING is written entirely in machine language and provides a simple means of avoiding garbage collection (those unnecessary delays that slow down your programs) With BUILD USING, you can choose how many digits should be displayed to right and left of the decimal point, and you can even fill the leading positions with the character of your choice For example, you can print the number '157.23' as '157.2', or '0000157.230', or '*****$157 AND 23/100 DOLLARS', or hundreds of other ways (including exponential formats). Working with strings is just as easy it's a snap to convert names from 'John' and 'Doe' to 'Doe, J.'. Also included are three levels of error trapping so you can trap and correct numbers or strings that cannot fit in your specified format

Utilities like BUILD USING are usually difficult to use because they must be located in one memory location (usually between DOS and the DOS file buffers) they cannot be used with your favorite editor or other special routines. BUILD USING does not have this limitation, as it can be easily located in many different memory locations: 1) the "normal" between DOS and DOS file buffers, 2) at HIMEM 3) APPENDED to your Applesoft program, or 4) anywhere else in memory Appending BUILD USING to your program is as simple as EXECing a TEXT file BUILD USING uses the "CALL" command thereby leaving the ampersand vector free for your own use

BUILD USING requires Applesoft in ROM (Language cards are fine) DOS 3.3 and a minimum of 32K only **$30.00**

Sensible Software

6619 Perham Drive Dept MO
West Bloomfield Michigan
48033 • (313) 399-8877
Visa and Mastercard
Welcome
Please add $1.25
postage and
handling per
diskette

68000 Instruction Set

by Joe Hootman

Editor's note: The author has compiled several detailed tables of 68000 instructions. This month's article offers a brief introduction to the 68000, along with table 1. We will publish more instruction tables in future issues, including those on addressing modes.

Since its introduction, the Motorola MC68000 has set a trend for 16-bit microprocessors that may become an industry standard. One reason for the 68000's quick success is the design philosophy Motorola used when the 68000 was conceived. Motorola wanted to give the 68000 a simple instruction set, and a flexible set of addressing modes. They also wanted it to be compatible with 6800 peripheral hardware and to contain many programming features. The implementation of this philosophy allows 6800/6809 microprocessor users to adapt easily to using both 68000 hardware and software. The instruction set, while compact, contains many special instructions that are unique to the 68000. These instructions allow two different processing states to exist in the 68000, as well as movement of data from 6800 peripherals. Special arithmetic instructions are also provided.

68000 users can turn to several references for help. The Motorola Users Manual[1] is essential for every 68000 user. An excellent book by L. J. Scanlon[2] explains the 68000 in detail, and provides excellent programming examples. The *68000 Microprocessor Handbook* by G. Kane[3] discusses characteristics and features of the 68000.

Special features of the 68000, which differentiate it from the 6800 series of microprocessors, are:

1. The data bus of the 68000 is 16 bits wide.
2. The 68000 is microprogrammed.
3. The 68000 is expandable. These expandable features will be defined by Motorola in the future.

***Table 1:* Data Movement**

<table>
<tr><th>Mnemonic</th><th>Data Size/CCR</th><th>Name</th><th>Comments/Address modes and opcode</th></tr>
<tr><td>MOVEA</td><td>16, 32
CCR<table><tr><td>X</td><td>N</td><td>Z</td><td>V</td><td>C</td></tr><tr><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td></tr></table></td><td>Move Address</td><td>Moves the contents of the source to the destination address register. Sign of word data is extended to 32 bits.
Opcode Format
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0<table><tr><td></td><td></td><td></td><td colspan="4">Destination</td><td colspan="2">Source</td></tr><tr><td>0</td><td>0</td><td>Size</td><td>Reg</td><td>0</td><td>0</td><td>1</td><td>Mode</td><td>Reg</td></tr></table>All addressing modes can be used for the source field. *</td></tr>
<tr><td>MOVEM</td><td>16, 32
CCR<table><tr><td>X</td><td>N</td><td>Z</td><td>V</td><td>C</td></tr><tr><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td></tr></table></td><td>Move Multiple Registers</td><td>Designated registers may be moved to an effective address.
Opcode Format
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0<table><tr><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>direction</td><td>0</td><td>0</td><td>1</td><td>size</td><td>Effective Mode</td><td>Addr. Reg.</td></tr><tr><td colspan="12">Register List Mask</td></tr></table>Direction: 0-register to memory,
1-memory to register
Size: 0-word field, 1-long word
Post Increment Address Mode<table><tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td>A7</td><td>A6</td><td>A5</td><td>A4</td><td>A3</td><td>A2</td><td>A1</td><td>A0</td><td>D7</td><td>D6</td><td>D5</td><td>D4</td><td>D3</td><td>D2</td><td>D1</td><td>D0</td></tr></table>Predecrement Addressing Mode<table><tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td>D0</td><td>D1</td><td>D2</td><td>D3</td><td>D4</td><td>D5</td><td>D6</td><td>D7</td><td>A0</td><td>A1</td><td>A2</td><td>A3</td><td>A4</td><td>A5</td><td>A6</td><td>A7</td></tr></table>Predecrement Address modes cannot use Addressing modes: 1, 2, 4, 11, 12, 13, 14 *
Post increment Address modes cannot use Address modes: 1, 2, 5, 13, 14 *</td></tr>
<tr><td>MOVEP</td><td>16, 32
CCR<table><tr><td>X</td><td>N</td><td>Z</td><td>V</td><td>C</td></tr><tr><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td></tr></table></td><td>Move Peripheral Data</td><td>Data is moved between data registers and memory. The high order byte of data bus is used when address is even; if address is odd the transfers are made to the low half of data bus. This instruction can be used to transfer data from 6800 devices.
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0<table><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>Data Register</td><td>Op-Mode</td><td>0</td><td>0</td><td>1</td><td>Address Register</td></tr><tr><td colspan="10">Displacement</td></tr></table>Opmode: 100-Word from memory to register
101-Long word from memory to register
110-Word from register to memory
111-Long word from register to memory</td></tr>
<tr><td>MOVEQ</td><td>32
CCR<table><tr><td>X</td><td>N</td><td>Z</td><td>V</td><td>C</td></tr><tr><td>-</td><td>*</td><td>*</td><td>0</td><td>0</td></tr></table></td><td>Move Quick</td><td>Move 8-bit immediate data to a data register, with sign extension.
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0<table><tr><td>0</td><td>1</td><td>1</td><td>1</td><td>Register</td><td>0</td><td>Data</td></tr></table></td></tr>
</table>

(Continued)

Mnemonic	Data Size/CCR	Name	Comments/Address modes and opcode
MOVE from SR	16 CCR X N Z V C - - - - -	Move from status register	The contents of the status register is moved to the destination location. 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 0 1 0 0 0 0 0 0 1 1 Effective Address Mode | Register This instruction cannot use the following Address modes: 2, 10, 11, 12, 13, 14.*
MOVE	8, 16, 32 CCR X N Z V C - * * 0 0	Move data from source to destination	This instruction moves the contents of the source to the destination. 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 0 0 Size Destination (EA) Register | Mode Source (EA) Mode | Register The size specifies the size of the data to be moved. 01-Byte 10-Long word 11-Word The effective address of the destination cannot use: 10, 11, 12, 13, 14* The effective address of the source can use all the addressing modes except 13, 14. *
MOVE to CCR	16 CCR X N Z V C * * * * *	Move condition code	The content of the source is moved to the condition code register. 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 0 1 0 0 0 1 0 0 1 1 Effective Address Mode | Register All addressing modes can be used except 13, 14*
SWAP	16 CCR X N Z V C - * * 0 0	Swap register halves	This instruction exchanges the upper half of a data register with the lower half of the data register. 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 0 1 0 0 1 0 0 0 0 1 0 0 0 Register The addressing mode of this instruction is inherent.
EXG	32 CCR X N Z V C - - - - -	Exchange registers	This instruction exchanges the contents of two registers. The exchange can take place between any of the registers. 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 1 1 0 0 Register Number 1 Op-mode Register Number Opmode: 01000-data registers 01001-address registers 10001-data and address registers (bits 0-2 must be address register number)
LEA	32 CCR X N Z V C - - - - -	Load effective address	The effective address is loaded into the specified address register. 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 0 1 0 0 Register 1 1 1 Effective Address Mode | Register This instruction cannot be used with the following addressing modes: 1, 2, 4, 5, 13, 14*
PEA	36 CCR X N Z V C - - - - -	Push effective address	The effective address is computed and the long word is pushed onto the stack. 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 0 1 0 0 1 0 0 0 0 1 Effective Address Mode | Register This instruction cannot use the following addressing modes: 1, 2, 4, 5, 13, 14 *

*The Addressing Modes will be presented in future issues.

4. The 68000 will work with 6800 parts, with a special software statement, and hardware for completing the handshaking requirements.

5. The 68000 is capable of addressing a large amount of memory.

6. The 68000 has a minimum number of instructions with a large number of addressing modes.

7. The 68000 is capable of supporting multiple users.

8. A supervisory state exists that is protected from access by the user. This state is used to support the overhead for multiple-user hardware interrupts, and certain classes of user-generated errors and traps.

9. The hardware architecture and software of the 68000 will easily support higher-level languages such as ADA, Pascal, Fortran, etc.

These features indicate that the 68000 will exist in its present form, or in a mildly modified form, for a significant length of time. The expandable features of the 68000 will allow users to learn a machine, and develop software and hardware, without fear of their work becoming quickly obsolete.

MOVE

Table 1 displays the MOVE instructions implemented in the 68000. These instructions move data both inside the processor from register to register, and from external RAM and external devices. The table indicates which bits are set in the USR with the completion of the instruction. The user part of the status register is the lower eight bits of the register; the system part is the upper eight bits. Motorola, in their instruction set, has chosen to refer to the user portion of the status register as the Condition Code Register (CCR).

A dash (—) in a bit location indicates that the operation has no effect on the bit. A * indicates that the operation may alter the bit, depending on the data and operation. Note that all the move operations except one do not set the V or C bits; these bits can only be set or cleared by an operation on data, or specific operations on the CCR.

The MOVE to CCR (move to condition code) can set or clear every bit in the CCR. This instruction could be used to check on the state of the first four bits of a word, or to set or clear bits in the CCR. (The instruction could also be used to unstack the CCR from memory.)

MOVE (MOVE data from source to destination) is a popular instruction because it allows movement of data between data registers and address registers inside the 68000. It also allows the movement of data between the 68000 and RAM.

MOVE and MOVE A (Move Address) are similar, except that MOVE A is the instruction reserved for moving data to the address registers. That is, the destination register in the MOVE A instructions are the address registers. The MOVE instruction is generally used to move data to and from the data registers. Proper coding results in the MOVE A instruction.

If you want to examine the status register to find out the state of the X, N, Z, V, and C bits, as well as allowing the checking of the interrupt mask, the trace (T), and supervisor status, use the MOVE from SR (Move from Status Register) instruction.

One of the most powerful move instructions is MOVE M (Move Multiple Registers). This instruction allows the movement of the selected data registers to consecutive memory locations. It could be used to store the results of multiple word arithmetic operations. For example, if the IEEE floating point arithmetic standard is implemented, the registers would contain the various components of the floating point number. Note that because both data registers and address registers can be moved, this instruction can be used to stack the registers.

MOVE P (Move Peripheral Data) is used to move 8-bit data to the data registers from memory, or transfer 8-bit data from the data registers to memory. This instruction, in conjunction with the proper hardware interface, allows the transfer of data between the 68000 and 6800 peripheral parts.

MOVE Q (Move Quick) allows the immediate movement of 8-bit data to any designated data register. The sign of the data is extended to fill the 32 bits of the data register, thus the operation is always a long word operation. Some assemblers will use a MOVE Q instruction every time 8-bit data are to be moved.

The SWAP (Swap Register Halves) is not the same as the previously discussed MOVE instructions, but it is a data movement instruction. This instruction will place the upper 16 bits of the designated data register into the lower 16 bits of the designated data register. The lower 16 bits will be moved to the upper 16 bits of the register. This instruction could be used to move data in a 32-bit register into a position to be tested by the CCR.

EXG (Exchange Registers) exchanges the contents of two registers using the full 32 bits of the two registers. EXG will exchange data registers with data registers, address registers with address registers, and address registers with data registers. This type of instruction could move an address register to a data register to allow operations and testing of the state of the address, while not destroying the contents of the data register.

LEA (Load Effective Address) loads a specified address register with an effective address. LEA loads all 32 bits of the address register from memory. For example, LEA cannot be used to load the address register from a data register. MOVE A can be used for this. LEA effectively increments or decrements the contents of a register. To implement the incrementing of a register, LEA is used with auto-incrementing or auto-decrementing addressing.

PEA (Push Effective Address) pushes the specified address onto the stack. This instruction is used as the traditional push instruction.

References

1. *16-Bit Microprocessor User's Manual*, Third Edition. Prentice-Hall, Inc.; Englewood Cliffs, New Jersey 07632.

2. *The 68000: Principles and Programming*, L.J. Scanlon, Howard W. Sams and Co. Inc.; 4300 West 62nd St., Indianapolis, Indiana 46268.

3. *68000 Microprocessor Handbook*, G. Kane, OSBORNE/McGraw-Hill; 630 Bancroft Way, Berkeley, California 94710.

Professor Hootman can be contacted at the University of North Dakota, Dept. of Electrical Engineering, University Station, Grand Forks, North Dakota 58202.

Wizard-SOB™

SERIAL OUTPUT BUFFER CARD

Input a new task while your old task prints out

NO MORE DELAY FOR SLOW PRINTERS

WIZARD-SOB is a serial printer interface that lets you operate your Apple while data from your previous operation is printing out. It ends waiting while your printer prints. The WIZARD-SOB contains a 16K byte "character buffer" that may be expanded to 32K bytes of character buffering. It accepts data at the Apple's rapid transfer rate, stores the data and feeds it to your printer at the rate the printer can accept.

- Fully compatible with all Apple II and Apple II Plus* computers
- Fully compatible with most Apple software including Applesoft BASIC*, Integer BASIC*, CP/M, Pascal, etc.
- Fully compatible with most Apple II expansion boards.
- Low power consumption for cool reliable operation
- Leads soldered directly into board for reliable operation
- Many formatting features.
- Two-year factory replacement warranty
- RS-232 printer interface compatible

Backed by the world's #1 independent manufacturer of peripheral controllers

Available at all fine Computer Stores $249.00

Telephone (800) 854-8737
(714) 850-1666

WESPER MICRO™ Systems
SUBSIDIARY OF WESPERCORP

*Registered Trademark of Apple Computers, Inc.

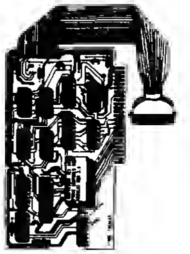
INPUT YOUR NEXT TASK WHILE YOUR PRINTER PRINTS OUT YOUR LAST TASK . . . WESPERMICRO SYSTEMS ADDS VALUE TO YOUR APPLE.

APPLE HARDWARE

201 838-9027 NEW

THE TACKLER™

DUAL • MODE PARALLEL INTERFACE FOR THE APPLE®
2 BOARDS IN ONE FOR NO MORE COMPATIBILITY PROBLEMS!

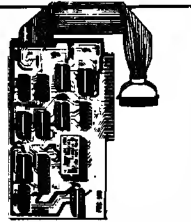


An intelligent board to provide easy control of your printer's full potential. Plus a standard parallel board at the flip of a switch -- your assurance of compatibility with assentially all software for the APPLE®. Hires printing with simple keyboard commands that replace hard to use software routines. No disks to load. Special features include inverse, doubled, and rotated graphics and many text control features, available through easy keyboard or software commands.

It's simple to print HIRES graphics from an APPLE computer with The Tacklar from TYMAC. This is the first truly universal parallel interface! Plus the ROM for your specific printer. Sophisticated intelligence when you need it, total compability that never lets you down.

Change printers -- no need to buy another board. Just plug in one of our ROM'S and you're all set. ROM'S available for Epson, C. Itoh, NEC, and Okidata -- others available soon.

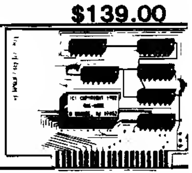
You've asked us to make the TACKLER better than the others and we did. $159.00 *



UPGRADEABLE PARALLEL PRINTER CARD

A Universal Centronics type parallel printer board complete with cable and connector. This unique board allows you to turn on and off the high bit so that you can access additional features in many printers. Easily upgradeable to a fully intelligent printer board with graphics and text dumps. Use with EPSON, C. ITOH, ANADEX, STAR-WRITER, NEC, OKI and others with standard Centronics configuration. **$139.00**

THE PERFORMER PRINTER FORMATTER BOARD for Epson, OKI, NEC 8023, CITOH 8510 provides resident HIRES screen dump and print formatting in firmware. Plugs into Apple slot and aasy access to all printer fonts through menu with PR# command. Uae with standard printer cards to add intelligence. **$49.00** specify printer.



THE MIRROR FIRMWARE FOR NOVATION APPLE CAT II®

The Data Communication Handler ROM Emulates syntax of an other popular Apple Modem product with improvements. Plugs directly on Apple CAT II Board. Supports Videx and Smarterm 80 column cards, touch tone and rotary dial, remote terminal, voice toggle, easy printer access and much more.

List $39.00 Introductory Price **$29.00**

DOUBLE DOS Plus

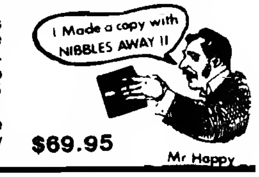
A piggy-back board that plugs into the disk-controller card so that you can switch select between DOS 3.2 and DOS 3.3. DOUBLE DOS Plus requires APPLE DOS ROMS. **$39.00**

SOFTWARE

NIBBLES AWAY II

AGAIN! Ahead of all others.

- **AUTO-LOAD PARAMETERS** . . . Free's the user from having to Manually Key in Param values used with the more popular software packages available for the Apple II.
- **EXPANDED USER MANUAL** . . . incorporates new Tutorials for all levels of expertice; Beginners Flowchart for 'where do I begin' to 'Advanced Disk Analysis' is included.
- **TRACK/SECTOR EDITOR** . . . An all new Track/Sector Editor, including the following features: Read, Write, Insert, Delete Search, and impreasive Print capabilities!
- **DISK DIAGNOSTICS** . . . Checks such thinga as: Drive Speed, Diskette Media Reliability, and Erasing Diskettes.
- **HIGHEST RATED** . . . Best back up Program in Softalk Poll (Rated 8.25 out of 10).
- **CONTINUAL UPDATES** . . . Available from Computer Applications and new listings on the source.



$69.95

Super PIX HIRES SCREEN DUMP — — —

The Software package that will allow your printer to dump page 1 or page 2 of the Apple Hires screen horizontally or vertically. Use with EPSON® MX-80 with or without GRAFTRAX® Roms, MX-70 – OKI® Microline 80, 82, 83, 82A, 83A – C. ITOH® 8510 and NEC 8023A. Requires Tymac Parallel Printer Board PPC-100 . . . **$24.95.**

APPLE LINK – A versitile modem utility that provides the Apple user the ability to transfer disk files and software over the phone. Only one package needed for full transfers. Compatable with all DOS file types. **$59.00**
(requires Hayes Micro Modem)

THE APPLE CARD – Two sided 100% plastic reference card Loaded with information of interest to all Apple owners. **$3.98**

VISA MasterCard

MICRO-WARE DIST. INC.
P.O. BOX 113
POMPTON PLAINS, N.J. 07444

Dealer and Distributor Inquires Invited.

It Pays to Write for MICRO!

MICRO Magazine is actively seeking articles to present to our unique audience of sophisticated computerists. We welcome contributions on any aspect of 6502/6809/68000 hardware and software. Send us your latest ready-to-key-in program!

Each issue of **MICRO** presents articles on a wide range of systems and topics. In addition, each issue includes a special feature. We are planning features on the following topics:

January-Math and Simulations   February-Languages   March-Printers
April-Communications   May-Wave of New Computers   June-Operating Systems

If you've developed material relating to these subjects, we'd like to see it! Send articles to the attention of the editorial department. We are continually looking for good articles and/or programs for the:

Atari, Apple, CBM/PET, VIC, 68000, 6809, and TRS-80 Color Computer.

Standard **MICRO** categories that may be included in any issue are:

I/O • BASIC Aids • Machine-Language Aids • Graphics
Games • Applications • Programming Techniques • Utilities
Programming Languages • Hardware

We are also interested in topics our readers feel are important. Send in your own feature suggestions!

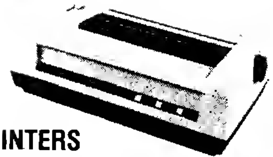
MICRO is sold in computer stores and on newsstands all over the world! Publishing an article in **MICRO** means quick distribution of your ideas.

We're proud of the fact that **MICRO**'s circulation is growing by leaps and bounds. And, we pay as well as BYTE! Get a copy of our Writer's Guide now!

Special Note to all MICRO authors: please begin sending your text as well as your programs in machine-readable format. This will enable us to edit the text with the help of word processing. Continue to include a printed text version as you include a printed program listing.

FOR YOUR APPLE II

Industry standard products at super saver discount prices



PARALLEL PRINTERS
NEC 8023 or C-ITOH 8510

(Virtually identical) Specifications: • 100 CPS dot matrix printer • 80 column print–136 characters per line • Tractor/friction feed • 7 different print fonts included • 2K printer buffer • Proportional spacing • Bit image graphics and graphic symbols.

NEC 8023 or C-ITOH	**$495**
NEC 8023 or C-ITOH 8510 with Parallel Interface and Cable	**$550**
EPSON 100 with Parallel interface and Cable	**$749**

Z-80 CARD FOR YOUR APPLE
MICROSOFT SOFTCARD

With CP/M® and MBASIC.

(List: $399) **$289**

NEW!

Best Buy!!!

ADVANCED LOGIC SYSTEM Z-CARD With C-PM®

Has everything the Softcard has except MBASIC. Works with Microsoft's disks too.

(List $269) ......**Special at $195**

NEW!

ALS SYNERGIZER

CP/M® operating package with an 80 column video board, CP/M® interface, and 16K memory expansion for Apple II. Permits use of the full range of CP/M® software on Apple II. Includes SuperCALC.

(List: $749) **$549**

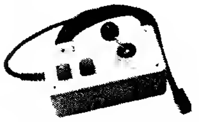
NEW!

U-Z-80 PROCESSOR BOARD
(From Europe)

Software compatible with Softcard and ALS Software **$149**

MICROSOFT + PREMIUM SYSTEM

Includes Videx Videoterm, Softswitch, Microsoft and Softcard, Microsoft and Z-80 Card, and Osborn CP/M® Manual **$595**



JOYSTICK

Takes the place of two Apple Paddle Controllers.

From BMP Enterprises. Heavy duty industrial construction and cable. Non-self centering. With polarity switches for consistent motion control.

(List: $59) **$39**

MONITORS FOR YOUR APPLE

AMDEK 300G (18MHZ Anti-Glare Screen)	**$179**
NEC 12" HIRES GREEN	**$179**

SUPER SPECIAL!

SPECIAL 12" GREEN MONITOR	**$99**

SPECIAL AND NEW

5 MEGABYTE HARD DISK

For Apple II. Supplied with controller. Use with CP/M, Apple DOS, & Apple Pascal **$1995**

5¼" DISK DRIVE

Use with standard Apple II disk controller. .**$295**

5¼" FLOPPY DISKS

With hub rings. Box of 10.

With other purchase	**$19.95**
Without purchase	**$23.00**

16K MEMORY EXPANSION MODULE

The preferred 16K RAM Expansion Module from PROMETHEUS. Fully compatible with CP/M® and Apple Pascal®. With full 1-year parts and labor warranty. (List: $169) **$75**

WORD PROCESSING SPECIAL WITH WORDSTAR AND SUPERCALC!

Do professional word processing on your APPLE. All necessary hardware and software included. Complete 80 column video display, enhanced character set, 16K memory board, Z-Card with CP/M® software, Wordstar and word processing software and SuperCALC.

(List: $1,128) ....**Special at $695**

NEW!

from Prometheus! **ExpandaRAM**

The only 128K RAM card that lets you start with 16K, 32K, or 64K of memory now and expand to the full 128K later. Fully compatible with Apple Pascal, CP/M®, and Visacalc. No Apple modification required. Memory management system included with all ExpandaRAMs. Disk emulators included with 64K and 128K versions.

MEM-32 Two rows of 16K RAMS make a 32K RAM Card	**$209**
MEM-64 One row of 64K RAM. With DOS 3.3 disk emulator	**$299**
MEM-128 Two rows of 64K RAMS installed make a 128K Card. With DOS 3.3 disk emulator	**$399**
MEM-RKT 64K RAM Add-On-Kits– 64K Dynamic RAMS. Each	**$125**
VISICALC Expansion Program for MEM-128	**$75**
MEM-PSL Pascal disk emulator for MEM-128	**$45**

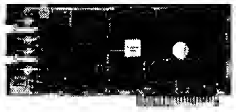
MODEMS FOR YOUR APPLE II

HAYES Smartmodem	**$229**
MICROMODEM II	**$279**

VERSAcard FROM PROMETHEUS

Four cards on one! With true simultaneous operation. Includes: (1) Serial Input/Output Interface, (2) Parallel Output Interface, (3) Precision Clock/Calendar, and (4) BSR Control. All on one card. Fully compatible with CP/M® and Apple Pascal®.

(List: $249) **$169**



80 COLUMN VIDEO DISPLAYS FOR APPLE II
SMARTERM

(Not to be confused with SUPRTERM)

Software switching from 80 to 40 and 40 to 80 characters. 9 new characters not found on the Apple keyboard. Fully compatible with CP/M® and Apple PASCAL®. With lowest power consumption of only 2.5 watts.

(List: $345) **$225**

SMARTERM EXPANDED CHARACTER SET

7" x 11" matrix with true decenders. Add to above **$40**

Best Buy! **Combination SMARTERM and EXPANDED CHARACTER SET**

Special at	**$260**
VIDEX, VIDEOTERM	**$249**
VIDEX ENHANCER II	**$119**

NEW!

CENTRONICS COMPATIBLE PARALLEL INTERFACE

From PROMETHEUS. For use with Epson, NEC, C-ITOH, and other printers. Fully compatible with CP/M® and Apple Pascal®.

PRT-1, Only **$69**

GRAPHITTI CARD

Prints HIRES page 1 or 2 from onboard firmware. Features: True 1:1 aspect ratio, prints emphasized mode, reverse mode, rotates 90 degrees . . . plus more. Compare all this with the Grappler. We think you'll agree that this is the best graphics card on the market. Specify for use with EPSON, NEC-8023, C-ITOH Prowriter, or Okidata.

(List: $125) **$89**

SOFTWARE

WORDSTAR	**Special at $195**
SPELLSTAR	**$125**
SUPERCALC	**$175**
D BASE II	**$525**
VISICALC	**$149**
DB MASTER	**$189**

All equipment shipped factory fresh Manufacturers' warranties included. Please add $3 00 per product for shipping and handling. California: add 6% tax. BART Counties: 6½%.

All items are normally in stock

Phone for Quick Shipment!

(415) 490-3420

. . . And we'll be here to help after you receive your order. Feel free to call the SGC Technical Staff for assistance.

SGC

The mail order specialists

342 Quartz Circle, Livermore, CA 94550

MICRO™

Apple Slices

By Tim Osborn

This month's Apple Slices presents a technique to quickly access records on a random access file. This technique, called hashing, key-to-address transformation, or randomizing, converts a user key into an address (or record number) where the record associated with the key is stored. This key may be a customer number, part number, invoice number, or anything that identifies the target record to the user.

Hashing should be used where quick random access is desirable. If the application calls for a lot of sequential processing, hashing will not be a good design choice for data organization. In situations where data is accessed by key in some undetermined order, hashing is at an advantage, because most records can be found within a few reads. Listing 2 is an example in which name and address records are hashed on the name field. I will refer to the listing to help clarify some points and to present a practical example.

An important consideration in hashing is dealing with collisions or overflows. An overflow occurs when two or more keys in the key range generate the same address (home record) within the address range. This can happen if there are as many variations in the key set as there are addresses to store the data. For example, in my name and address retrieval system (listing 2), the file can hold up to a maximum of 500 records (a system design constraint) and there are an infinite number of names. Therefore, many names may be transformed into the same address, creating overflow situations.

Some overflow handling techniques include:

1. *Overflow linked lists:* A separate area of the file is set aside for overflows. If an overflow occurs, it is stored in the first available overflow space and chained off from the home record. On a recall it is necessary to search the chain until the key of the stored record matches the search key. The disadvantage of this technique is that the chain must be maintained and the overflow space must be managed, which adds complexity. The advantage is that any number of overflows can be supported as long as there is spare room in the overflow area.

2. *Distributed overflow space:* Instead of using a separate overflow area, overflows are interleaved with normal records by setting aside some records and designating them as overflow. The advantage here is that the proximity of an overflow may allow it to be accessed without requiring additional head movement of the random access device. For Apple owners this means that we would attempt to store overflows on the same track as the home record. The disadvantage is that chain maintenance adds complexity to the system.

3. *Open addressing:* This technique tries to store the record in the home space first; if it is full, it stores the record in the next available space. This method is easy to implement. As long as a sufficient amount of free space exists in the file, the record can be stored close to the home record. The disadvantage is that, when the file begins to fill up, the search time increases, as it takes longer to find an unused record.

Listing 1

```
0 A$ = ""
10 CD$ =  CHR$ (4)
20  PRINT CD$"OPEN TEST-FILE,L100"
30  FOR J = 0 TO 500
40  PRINT CD$"WRITE TEST-FILE,R";J
50  FOR I = 1 TO 6
60  PRINT A$
70  NEXT
80  NEXT
90  PRINT CD$"CLOSE TEST-FILE"
```

After considering the various techniques available, I chose the open addressing technique because of its design simplicity. I implemented this in my FIND routine (lines 9000-9060) by incrementing the record number (KEY) until a free record was found or until I searched 20 records. I recognize a free record by NM$ being equal to NAM$ (NAM$ is set to null by the ADD routine before calling FIND), or by NM$ being equal to "0". NM$ is null if the record has never been used (this is the way the file is initialized when listing 1 creates the file) and NM$ is equal to "0" if it was used at one time but is now available. The reason for a different code for a once-used record is that the program must continue to search up to 20 records, or until NM$ is equal to null (" "), to make sure the new record is not a duplicate. If NM$ = " ", then I know a record with this same key could not have been stored further into the file. It would have been stored here if it had searched this far before. If NM$ = "0", it is possible that this record was used when a record with the new record's key was added and the system had to look further into the file at that time to find a free space.

If it finds a duplicate, the RC (return code) is set to 3 and returns to the ADD subroutine. The ADD subroutine will display a message and return to the main menu (line 7034). If NM$ = "0", FIND saves the available record number in KY and sets RC = 2. This tells the ADD subroutine to use KY instead of KEY as the available space. This is necessary because the FIND subroutine continues to increment KEY to search for a record with a null value in NM$. This process can be described as follows (upon return to ADD):

1. If RC = 0, then KEY = the available space where we can store the new record.

2. If RC = 1, then 20 records were searched and all were used. (ADD will display a "FILE TOO FULL" message.)

3. If RC = 2, then a previously used record was available. The value in KY will be used as the available space.

4. If RC = 3, then the new record is a duplicate. (ADD will display a "DUPLICATE RECORD" error message.)

FIND can be forced to search more records by raising the value of BIG (which is currently set at 20). If BIG is decremented, it is possible that previously stored records will not be found. I did some extensive testing with random key values and found that the "FILE TOO FULL" message was produced when the file contained approximately 300 records with BIG set at 20. The trade-off is between efficiency and space; the further you allow records to be stored from the home record, the lower the efficiency. The more you limit record storage to the home record, the larger the amount of wasted space.

The other major consideration when designing a hashing system is the key-to-address transformation technique. This technique converts the user key into a physical location for storage. All techniques transform the key into a number within an order of magnitude that is the same as the number of addresses (records). Then they multiply the intermediate result by a factor that limits the final result to the range of addresses available on the storage device.

In my system I chose a method called center-squaring. This method involves taking the key value, converting it into a number which can be squared, squaring the number, stripping off as many outer digits as necessary, and multiplying by a factor to limit the address to the valid range. My COMPUTE ADDRESS subroutine (lines 6000-6448) follows this sequence of operations.

1. Line 6000 sets KEY to 0. KEY will become the address of the home record — the first record where we will attempt to store a new name and address record.

2. Line 6002 finds the length of KEY$, which holds the name associated with the new name and address record.

3. Lines 6005-6030 convert KEY$ into a number through table lookups and store the accumulation in KEY. The (J * J / 3) expression in line 6020 helps insure that a number of the proper magnitude is produced.

Listing 2

```
1  GOSUB 10000: REM  INITIALIZE TABLE
10 BIG = 15:FL = 500:CD$ =  CHR$ (4): DIM SV$(6)
14  PRINT CD$"OPEN TEST-FILE,L100"
15  PRINT CD$
16  IF RC = 0 THEN  GOTO 18
17  FOR J = 1 TO 3000: NEXT J:RC = 0
18  HOME : PRINT "(Q)UIT (A)DD"
20  INPUT "(C)HANGE (D)ELETE (I)NQUIRE ";A$
30  IF A$ = "A" THEN  GOSUB 7000: GOTO 16
40  IF A$ = "C" THEN  GOSUB 7200: GOTO 16
50  IF A$ = "D" THEN  GOSUB 7500: GOTO 16
60  IF A$ = "I" THEN  GOSUB 8500: GOTO 16
70  IF A$ = "Q" THEN  PRINT CD$"CLOSE TEST-FILE": END
80  GOTO 18
5998  REM
5999  REM  ****COMPUTE ADDRESS****
6000 KEY = 0
6002 LN =  LEN (KEY$)
6005  FOR I = 1 TO LN
6010 MD$ =  MID$ (KEY$,I,1)
6015  FOR J = 0 TO 35
6020  IF MD$ = TBL$(J) THEN KEY = KEY + (J * J / 3):J = 35
6030  NEXT : NEXT
6050 KEY = KEY * KEY
6060 KEY$ =  STR$ (KEY)
6070 KEY$ =  MID$ (KEY$,2,3)
6080 KEY =  VAL (KEY$)
6090 KEY =  INT (KEY * .5)
6100  RETURN
6448  REM
6449  REM  ****WRITE RECORD****
6500  PRINT CD$;"WRITE TEST-FILE ,R";KEY
6510  PRINT NM$
6520  PRINT RD$
6525  PRINT CTY$
6527  PRINT ST$
6530  PRINT ZP$
6535  PRINT TEL$
6540  PRINT CD$
6550  RETURN
6998  REM
6999  REM  ****ADD RECORDS****
7000 FUNC$ = "ADD":NAM$ = ""
7005  GOSUB 8000: REM  GET INPUTS
7006 SV$(1) = NM$:SV$(2) = RD$:SV$(3) = CTY$:SV$(4) = ST$:SV$(5) = ZP$:SV
     $(6) = TEL$
7007 KEY$ = NM$
7010  GOSUB 6000: REM  COMPUTE ADDRESS
7020  GOSUB 9000: REM  FIND AVAIL. REC
7025 FUNC$ = ""
7027  IF RC > 0 THEN  PRINT CD$
7030  IF RC = 1 THEN  INVERSE : PRINT "FILE TOO FULL": NORMAL : RETURN
7032  IF RC = 2 THEN KEY = KY
7034  IF RC = 3 THEN  INVERSE : PRINT "DUPLICATE RECORD": NORMAL : RETURN
7035 NM$ = SV$(1):RD$ = SV$(2):CTY$ = SV$(3):ST$ = SV$(4):ZP$ = SV$(5):TE
     L$ = SV$(6)
7040  GOSUB 6500: REM  WRITE RECORD
7050  RETURN
7198  REM
7199  REM  ****CHANGE*****
7200  HOME : PRINT "DO YOU WISH TO CHANGE THE NAME?"
7210  INPUT "REPLY (Y)ES OR (N)O ";A$
7220  IF A$ = "N" GOTO 7290
7230  IF A$ <  > "Y" GOTO 7210
7235  INPUT "OLD NAME = ";NM$: IF NM$ = "" GOTO 7235
7240 KEY$ = NM$: GOSUB 6000: REM  COMPUTE ADDRESS
7250 NAM$ = NM$: GOSUB 9000: REM  FIND RECORD
7260  IF RC = 1 THEN : INVERSE : PRINT "RECORD NOT FOUND": NORMAL : PRINT
     CD$: RETURN
7270 NM$ = "0": GOSUB 6500: REM DELETE OLD RECORD
7280  INPUT "NEW NAME = ";NM$: IF NM$ = "" GOTO 7280
7282 KEY$ = NM$
7284  GOSUB 6000: REM  COMPUTE NEW KEY
7286  GOTO 7330
7290  INPUT "NAME     = ";NM$:KEY$ = NM$: IF NM$ = "" GOTO 7290
7300  GOSUB 6000
7310 NAM$ = NM$: GOSUB 9000: REM  FIND RECORD
7320  IF RC = 1 THEN  INVERSE : PRINT "RECORD NOT FOUND": NORMAL : PRINT
     CD$: RETURN
7330  INPUT "CHANGE STREET? ,(Y)ES OR (N)O ";A$
7340  IF A$ = "Y" THEN  INPUT "STREET   = ";RD$: GOTO 7360
7350  IF A$ <  > "N" GOTO 7330
```

(Continued)

Listing 2 *(Continued)*

```
7340  IF A$ = "Y" THEN  INPUT "STREET   = ";RD$: GOTO 7360
7350  IF A$ <  > "N" GOTO 7330
7360  INPUT "CHANGE CITY?, (Y)ES OR (N)O ";A$
7370  IF A$ = "Y" THEN  INPUT "CITY     = ";CTY$: GOTO 7382
7380  IF A$ <  > "N" GOTO 7360
7382  INPUT "CHANGE STATE?, (Y)ES OR (N)O ";A$
7384  IF A$ = "Y" THEN  INPUT "STATE    = ";ST$: GOTO 7390
7386  IF A$ <  > "N" GOTO 7382
7390  INPUT "CHANGE ZIP?, (Y)ES OR (N)O ";A$
7400  IF A$ = "Y" THEN  INPUT "ZIP      = ";A$: GOTO 7420
7410  IF A$ <  > "N" GOTO 7390
7420  INPUT "CHANGE TEL. NO?, (Y)ES OR (N)O ";A$
7430  IF A$ = "Y" THEN  INPUT "TEL. NO. = ";TEL$: GOTO 7450
7440  IF A$ <  > "N" GOTO 7420
7450  GOSUB 6500
7470  RETURN
7498  REM
7499  REM  ****DELETE****
7500  HOME
7510  INPUT "NAME    = ";KEY$
7515 NAM$ = KEY$
7520  GOSUB 6000: REM  COMPUTE ADDRESS
7530  GOSUB 9000: REM  FIND RECORD
7540  IF RC = 1 THEN  INVERSE : PRINT "RECORD NOT FOUND": NORMAL : PRINT
     CD$: RETURN
7550 NM$ = "0"
7560  GOSUB 6500: REM WRITE RECORD
7570  RETURN
7998  REM
7999  REM  ****GET INPUTS****
8000  HOME
8010  INPUT "NAME    = ";NM$: IF NM$ = "" OR NM$ = "0" GOTO 8000: REM   R
     ESERVED NAMES
8020  INPUT "STREET  = ";RD$
8030  INPUT "CITY    = ";CTY$
8035  INPUT "STATE   = ";ST$
8040  INPUT "ZIP     = ";ZP$
8050  INPUT "TEL. NO.= ";TEL$
8060  RETURN
8498  REM
8499  REM  ****INQUIRE****
8500  HOME : INPUT "NAME    = ";NAM$
8505 KEY$ = NAM$
8510  GOSUB 6000: REM  COMPUTE ADDRESS
8520  GOSUB 9000: REM  FIND RECORD
8530  IF RC = 1 THEN  INVERSE : PRINT "RECORD NOT FOUND": NORMAL : PRINT
     CD$: RETURN
8550  PRINT "STREET  = ";RD$
8560  PRINT "CITY    = ";CTY$
8565  PRINT "STATE   = ";ST$
8570  PRINT "ZIP     = ";ZP$
8580  PRINT "TEL. NO.= ";TEL$
8590  PRINT
8600  PRINT "PRESS RETURN TO CONTINUE"
8610  GET A$
8620  RETURN
8998  REM
8999  REM   ****FIND RECORD****
9000 J = 0:RC = 0
9010  IF J <  > BIG GOTO 9020
9012  IF RC = 0 THEN RC = 1: RETURN
9014  RETURN : REM  RC=2
9020  IF KEY > FL THEN KEY = 0
9030  PRINT CD$"READ TEST-FILE,R";KEY
9040  INPUT NM$,RD$,CTY$,ST$,ZP$,TEL$
9042  IF FUNC$ <  > "ADD" THEN  GOTO 9050
9044  IF NM$ = "0" AND RC = 0 THEN KY = KEY:RC = 2: GOTO 9060
9045  IF NM$ = SV$(1) THEN RC = 3: RETURN
9050  IF NM$ = NAM$ THEN  PRINT CD$: RETURN
9060 KEY = KEY + 1:J = J + 1: GOTO 9010
9998  REM
9999  REM  ****BUILD TABLE****
10000  DIM TBL$(35)
10010  FOR J = 0 TO 35: READ TBL$(J): NEXT J
10015  RETURN
10100  DATA  "1","2","3","4","5","6","7","8","9","0","Q","W","E","R","T",
     "Y","U","I","O","P","A","S","D","F","G","H","J","K","L","Z","X","C",
     "V","B","N","M"
```

4. Line 6050 squares KEY.

5. Line 6060 converts KEY back to a string (KEY$).

6. Line 6070 takes out the second, third, and fourth digits, and stores them back in KEY$.

7. Line 6080 converts KEY$ back to a numerical value (KEY).

8. Line 6090 brings KEY within the range of records on my file. When listing 1 creates the file, it creates 500 records, thus the factor of .5.

It is important that a key-to-address transformation method spread the records throughout the file as evenly as possible to minimize overflows. The address generated should bear no relation to the key, so that a key with a higher value may be stored below a key with a lower value. This is often necessary because applications have many records with similar keys. If there is a relationship between the keys and the addresses generated, it would cause one part of the file to become full; overflows will occur in great numbers.

I hope I have provided enough information to allow you to develop your own hashing applications. If you don't feel up to programming your own system, listing 1 and listing 2 can be used as a quick-access address book. Remember to run listing 1 first to intitialize the file. If you get the "FILE TOO FULL" error message, then you may wish to bump up BIG, which is initialized on line 10. This will slow down access but will allow you to store more records.

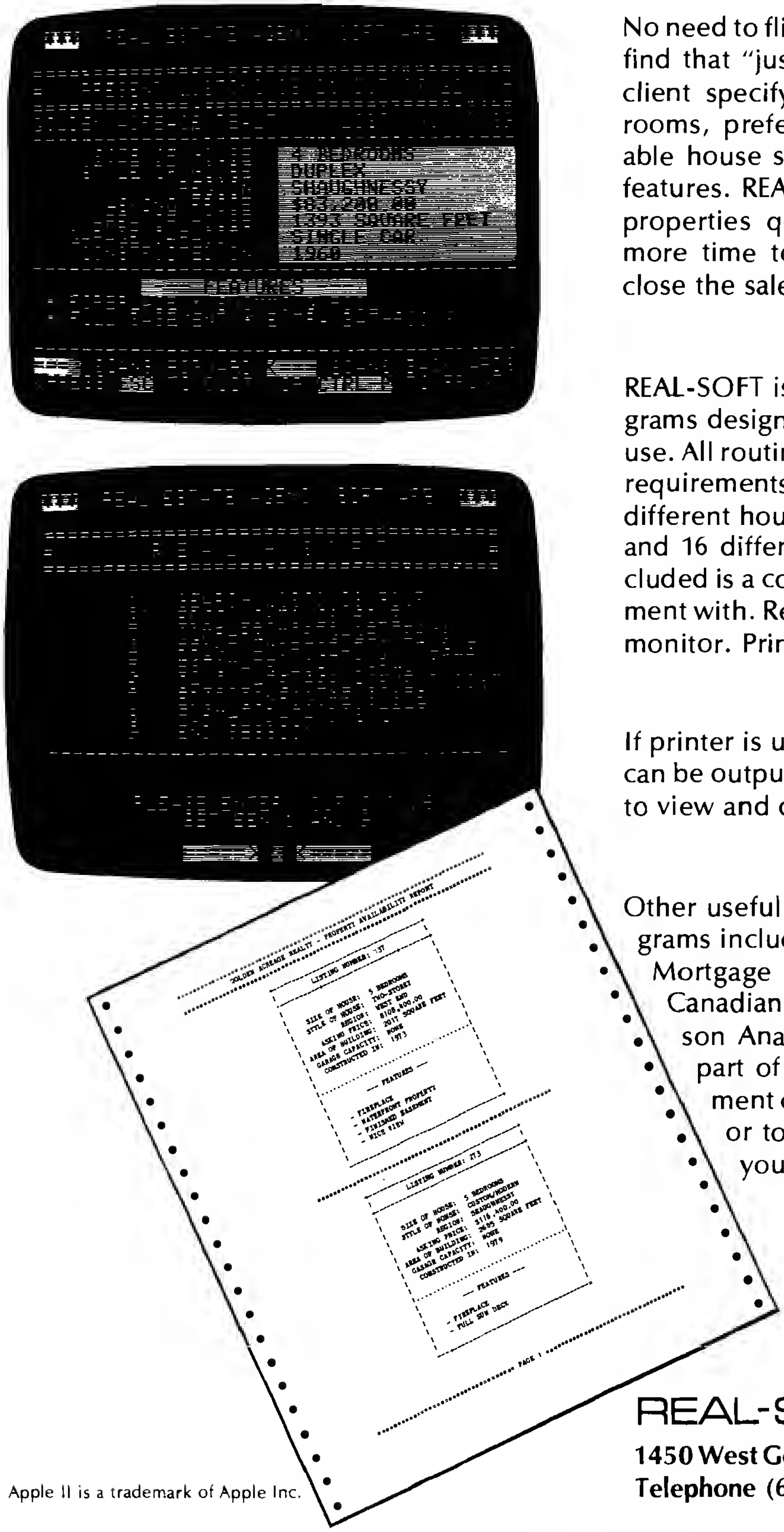
Otherwise, you can change listing 1 to load more than 500 records. If you do, remember to change FL (file length — line 10) from 500 to the number of records you wish to load. Also change the range multiplication factor from .5 to .xxx, where xxx is equal to the number of records you wish to load (line 6090). If you plan to load more than a thousand records, you will need to change the center-extraction from MID$ (KEY$,2,3) to MID$ (KEY$,2,4) or MID$ (KEY$,3,4).

Reference

1. *Computer Data-Base Organization*, by James Martin, Prentice-Hall, Englewood Cliffs, New Jersey.

Three minutes to find that "just right" property.

Search through up to 3600 property listings with Real-Soft programs for Real Estate Agencies.



```
3 BEDROOMS
DUPLEX
SHOUGHNESSY
$83,200.00
1393 SQUARE FEET
SINGLE CAR
1960

FEATURES
```

```
GOLDEN ACREAGE REALTY - PROPERTY AVAILABILITY REPORT

LISTING NUMBER: 137

   SIZE OF HOUSE:  5 BEDROOMS
  STYLE OF HOUSE:  TWO-STOREY
          REGION:  WEST END
    ASKING PRICE:  $108,800.00
AREA OF BUILDING:  2017 SQUARE FEET
GARAGE CAPACITY:   NONE
 CONSTRUCTED IN:   1973

        --- FEATURES ---

- FIREPLACE
- WATERFRONT PROPERTY
- FINISHED BASEMENT
- NICE VIEW

LISTING NUMBER: 273

   SIZE OF HOUSE:  5 BEDROOMS
  STYLE OF HOUSE:  CUSTOM/MODERN
          REGION:  SHAUGHNESSY
    ASKING PRICE:  $116,400.00
AREA OF BUILDING:  2695 SQUARE FEET
GASAGR CAPACITY:   NONE
 CONSTRUCTED IN:   1979

        --- FEATURES ---

- FIREPLACE
- FULL SUN DECK

PAGE 1
```

No need to flip through endless pages of listings to find that "just right" property. Simply have your client specify the price range, number of bedrooms, preferred geographical location, acceptable house styles and up to 12 additional listing features. REAL-SOFT programs will find the right properties quickly and impressively giving you more time to view the selected properties and close the sale!

REAL-SOFT is an integrated set of computer programs designed specifically for real estate agency use. All routines are fully customizable to meet the requirements of virtually any agencies. Up to 10 different house styles, 12 different listing features and 16 different areas. Very easy to learn — included is a complete set of practice data to experiment with. Requires 48K Apple II, 2 disk drives and monitor. Printer is optional.

If printer is used a printout of selected properties can be output in only a few minutes for your client to view and discuss in detail.

Other useful routines in this powerful set of programs include a Property Availability Report, full Mortgage Amortization Tables (American OR Canadian method), and a Mortgage Comparison Analysis Report. Then use still another part of REAL-SOFT to estimate the replacement cost of any listing (using local factors), or to estimate the mortgage amount that your client can qualify for!

Available at your local Apple dealer.

REAL-SOFT Real estate software

1450 West Georgia Street, Vancouver, B.C. V6G 2T8
Telephone (604) 669-2262

Apple II is a trademark of Apple Inc.

AMPER-POS for the Apple

by Philippe Francois

This machine-language utility extends Applesoft to include a POSition function similar to Pascal's.

AMPER-POS
requires:

Apple II or Apple II Plus
with Applesoft in ROM
or language card

To Pascal, there is a POS function that searches within the string variable TEXT for an occurrence of a character string matching the context of the "pattern" variable PATTERN. The value returned is an integer representing the first character position where the matching string is found. If no matching string is found, the POS returns a value of zero that does not correspond to one of the allowable character positions in a STRING variable (see reference 1).

```
PROGRAM POSITION;
VAR PATTERN, TEXT : string;
BEGIN
  PATTERN: = 'pal';
  TEXT: = 'So long,pal..''
  WRITELN('Position of ',PATTERN,'
    in ',TEXT,' = ',POS(PATTERN,
    TEXT));
END.
```

This function fails in Applesoft, so you need to write a short machine routine, AMPER-POS, which gives the first occurrence of PATTERN in TEXT, but also gives the other occurrences (if any).

The Routine

AMPER-POS was designed to find the occurrences of a sub-string A$ in a string B$. There are two entry points:

1 . The first entry point finds the first occurrence of A$ in B$ and has the format:

& P (A$,B$,C%)

where:

A$ is the sub-string variable
B$ is the string variable
C% is the output integer variable

2. The second entry point finds the other strings' occurrences and has the simple format:

CALL 880

System Requirement

AMPER-POS is written in assembly language and can be run on a 16K, 32K, or 48K system with either disk or cassettes. The $300 (768 decimal) address has been chosen for ORG because the $300,$3FF memory range is available for short machine-language programs (see reference 2).

Listing 1

```
0800           PAG
0800    ;
0800    ;          ****************************************
0800    ;          ***                                  ***
0800    ;          ***         POSITION FUNCTION        ***
0800    ;          ***            (AMPER-POS)           ***
0800    ;          ***                BY                ***
0800    ;          ***         PHILIPPE FRANCOIS        ***
0800    ;          ***                                  ***
0800    ;          ****************************************
0800    ;
0800    ;
0800    ; AMPERPOS IS AN APPLESOFT ROM ROUTINE FOR FINDING OCCURENCES
0800    ; OF A SUB-STRING "PATTERN" IN A STRING "TEXT"
0800    ;
0800    ; ENTRY POINTS:
0800    ;
0800    ;               1) FOR THE FIRST OCCURENCE: & P (A$,B$,C%)
0800    ; WHERE:
0800    ;               A$=SUB STRING (PATTERN)
0800    ;               B$=STRING (TEXT)
0800    ;               C%=POSITION OF A$ IN B$
0800    ;
0800    ; IT RETURNS:
0800    ; A) ZERO IN C% IF A$ IS NOT IN B$
0800    ; B) THE FIRST CHARACTER POSITION WHERE
0800    ;    THE MATCHING STRING A$ IS FOUND IN B$
0800    ;
0800    ;               2) CALL 880 FOR THE NEXT OCCURENCES OF A$ IN B$
0800    ;
0800    ;
0800    ; AMPERPOS USEAGE
0800    ;
0800    BEGINL EPZ $00             ; EFFECTIVE STRING BEGINNING
0800    BEGINH EPZ BEGINL+$01
0800    M      EPZ BEGINH+$01      ; "PATTERN" LENGTH
0800    N      EPZ M+$01           ; "TEXT" LENGTH
0800    PTRL   EPZ N+$01           ; PTRL,H IS A POINTER
0800    PTRH   EPZ PTRL+$01        ; TO THE STRING ADDRESS
0800    INDICJ EPZ PTRH+$01        ; STARTING POSITION IN TEXT
0800    FLAG   EPZ INDICJ+$01      ; IF PATTERN IS FOUND IN TEXT
0800    ;                          ; THEN FLAG=1
0800    ;                          ; ELSE FLAG=0
0800    ; APPLESOFT USEAGE
0800    ;
0800    XSAV   EPZ $46
0800    VALTYP EPZ $11             ; 0=NUMBER FF=STRING
0800    CHKOPN EQU $DEBB           ; CHECK THE "("
0800    CHKCLS EQU $DEB8           ; CHECK THE ")"
0800    CHKCOM EQU $DEBE           ; CHECK THE ","
0800    ERROR  EQU $DD76           ; "MISMATCH" ERROR
0800    CHRGET EQU $00B1
0800    PTRGET EQU $DFE3           ; READ A VARIABLE AND FIND IT IN
0800    ;                          ; MEMORY(ON EXIT THE ADDRESS TO
0800    ;                          ; THE VALUE OF THE VARIABLE IN A,Y)

0800    ;
0300           ORG $300
0300    ;
```

(Continued)

Entering AMPER-POS

To enter the program, type in the hex codes or instructions from the source listing with an assembler. To save the program, type:

```
BSAVE AMPER-POS OBJ,A$300,
   L$BC (for disk)

300-3BCW (for cassettes)
```

Theory

Suppose we want to find the first instance of the pattern string PATTERN(1)...PATTERN(M) in the target string TEXT(1)..TEXT(N),M =N. (PATTERN(I) denotes the Ith character of PATTERN.) An obvious approach is to align PATTERN with the leftmost M characters at TEXT and start matching. When a mismatch occurs, shift PATTERN right one character and restart matching from its first position. Continue until either PATTERN matches or there are no more characters to match in TEXT.

The following algorithm that we use in the AMPER-POS program is the formal version of what has just been described (see reference 3 for more details):

```
INTEGER i,j,m,n;
STRING pattern(1:m),text(1:n);
i: = j: = 1;
DO WHILE i < =m AND j < =n;
 IF pattern(i) = text(j) THEN
 DO/*keep matching*/
  i: = i + 1;/*bump pattern position*/
  j: = j + 1;/*and text position*/
 END
 ELSE
 DO/*backtrack*/
  j: = j + 2 – i;
  i: = 1;
 END
END
```

Program Logic

Here is a synopsis of the program logic:

Lines 64-143 retrieve the parameters from the "&" statement. Applesoft ROM routines are used to reduce the length of the program.

Lines 152-194 translate the precedent algorithm. Take the X and Y registers for the I and J indexes, but because ABSOLUTE,X and INDIRECT,Y addressing begin with the zero value for X and Y, the algorithm has been slightly modified.

Listing 1 *(Continued)*

```
0300          60   ; ************************************************
0300          61   ; *                    PARSING                    *
0300          62   ; ************************************************
0300          63   ;
0300          64   ; ----------------
0300          65   ;  VERIFY " P ( "
0300          66   ; ----------------
0300          67   ;
0300 C9D0     68          CMP #"P"              ; CHECK THE "P"
0302 F003     69          BEQ *+5
0304          70          PAG
0304 4C76DD   71          JMP ERROR
0307 20B100   72          JSR CHRGET
030A 20BBDE   73          JSR CHKOPN            ; CHECK THE "("
030D          74   ;
030D          75   ; ------------------
030D          76   ;  SEARCH "PATTERN"
030D          77   ; ------------------
030D          78   ;
030D 20E3DF   79          JSR PTRGET
0310 E611     80          INC VALTYP
0312 F003     81          BEQ STRG1             ; VALTYP EQUAL ZERO,IT IS A STRING
0314 4C76DD   82          JMP ERROR
0317          83   STRG1:
0317 8504     84          STA PTRL
0319 B405     85          STY PTRH
031B A000     86          LDY #$00
031D B104     87          LDA (PTRL),Y
031F 8502     88          STA M
0321 C8       89          INY
0322 B104     90          LDA (PTRL),Y
0324 8500     91          STA BEGINL
0326 C8       92          INY
0327 B104     93          LDA (PTRL),Y
0329 8501     94          STA BEGINH
032B          95   ;
032B          96   ; -------------------------------------------------
032B          97   ;  REWRITE THE SUBSTRING "PATTERN" AT PATTRN ADDRESS
032B          98   ; -------------------------------------------------
032B          99   ;
032B A0FF    100          LDY #$FF
032D C8      101   LABEL1 INY
032E B100    102          LDA (BEGINL),Y
0330 99BA03  103          STA PATTRN,Y
0333 C402    104          CPY M
0335 D0F6    105          BNE LABEL1
0337         106   ;
0337 20BEDE  107          JSR CHKCOM
033A         108   ;
033A         109   ; ---------------
033A         110   ;  SEARCH "TEXT"
033A         111   ; ---------------
033A         112   ;
033A 20E3DF  113          JSR PTRGET
033D E611    114          INC VALTYP
033F F003    115          BEQ STRG2
0341 4C76DD  116          JMP ERROR
0344         117   STRG2:
0344 8504    118          STA PTRL
0346 8405    119          STY PTRH
0348 A000    120          LDY #$00
034A B104    121          LDA (PTRL),Y
034C 8503    122          STA N
034E C8      123          INY
034F B104    124          LDA (PTRL),Y
0351 8500    125          STA BEGINL
0353 C8      126          INY
0354 B104    127          LDA (PTRL),Y
0356 8501    128          STA BEGINH
0358         129   ;
0358 20BEDE  130          JSR CHKCOM
035B         131   ;
035B         132   ; ------------------------
035B         133   ;  SEARCH OUTPUT VARIABLE
035B         134   ; ------------------------
035B         135   ;
0358 20E3DF  136          JSR PTRGET
035E A611    137          LDX VALTYP
0360 F003    138          BEQ INT
0362 2076DD  139          JSR ERROR
0365         140          PAG
0365         141   INT:
0365 8504    142          STA PTRL
0367 8405    143          STY PTRH
0369         144   ;
0369 20B8DE  145          JSR CHKCLS
036C         146   ;
036C         147   ; ************************************************
036C         148   ; *                   RESOLVING                   *
036C         149   ; ************************************************
036C         150   ;
036C         151   ; I:=REG. X
036C         152   ; J:=REG. Y
036C         153   ;
036C A900    154          LDA #$00              ; INDICJ:=0
036E 8506    155          STA INDICJ            ;
0370         156   ;
0370         157   ; --------------------------------------------
0370         158   ;  SECOND ENTRY POINT: CALL (ORG ADDRESS+$63)
0370         159   ; --------------------------------------------
```

(Continued,

Listing 1 *(Continued)*

```
0370 A200     161          LDX #$00          ; I:=0 ; J:=INDICJ
0372 8607     162          STX FLAG
0374 A406     163          LDY INDICJ
0376          164   ;
0376          165   BEGIN:
0376          166   ;
0376 E402     167          CPX M             ; DO WHILE I<=M AND J<=N
0378 F027     168          BEQ END           ;
037A C403     169          CPY N             ;
037C F023     170          BEQ END           ;
037E B100     171          LDA (BEGINL),Y    ;
0380 DDBA03   172          CMP PATTRN,X      ;    IF PATTERN(I)=TEXT(J) THEN
0383 D00E     173          BNE LABEL2        ;
0385 A507     174          LDA FLAG          ;       IF FLAG=0 THEN
0387 D006     175          BNE LABEL3        ;
0389 A901     176          LDA #$01          ;       DO
038B 8507     177          STA FLAG          ;          FLAG:=1
038D 8406     178          STY INDICJ        ;          INDICJ:=J
038F          179   ;                        ;       END
038F          180   ;                        ;       ELSE
038F          181   ;                        ;       DO
038F E8       182   LABEL3 INX               ;          I:=I+1
0390 C8       183          INY               ;          J:=J+1
0391          184   ;                        ;       END
0391 D0E3     185          BNE BEGIN         ;
0393 C8       186   LABEL2 INY               ;    ELSE
0394 8646     187          STX XSAV          ;    DO
0396 38       188          SEC               ;       J:=J+2-I
0397 98       189          TYA               ;
0398 E546     190          SBC XSAV          ;
039A A8       191          TAY               ;
039B A200     192          LDX #$00          ;       I:=1
039D 8607     193          STX FLAG          ;       FLAG:=0
039F          194   ;                        ;    END
039F F0D5     195          BEQ BEGIN         ;
03A1          196   ;                        ; END
03A1          197   ;
03A1          198   ; ----------------
03A1          199   ;  END OF ALGORITM
03A1          200   ; ----------------
03A1          201   ;
03A1          202   END:
03A1 A507     203          LDA FLAG          ;
03A3 D004     204          BNE LABEL4        ; IF FLAG=0 THEN
03A5 A9FF     205          LDA #$FF          ; INDICJ:=0
03A7 8506     206          STA INDICJ        ;
03A9          207   ;
03A9          208   LABEL4:
03A9 A506     209          LDA INDICJ        ; ELSE
03AB          210          PAG
03AB 18       211          CLC               ; INDICJ:=INDICJ+1
03AC 6901     212          ADC #$01
03AE          213   ;
03AE          214   ; ------------------------------
03AE          215   ;  PUT INDICJ IN OUTPUT VARIABLE
03AE          216   ; ------------------------------
03AE          217   ;
03AE A001     218          LDY #$01
03B0 9104     219          STA (PTRL),Y
03B2          220   ;
03B2          221   ; ----------------------------------------
03B2          222   ;  FOR NEXT OCCURENCE : INDICJ:=INDICJ+M
03B2          223   ; ----------------------------------------
03B2 18       224          CLC
03B3 A506     225          LDA INDICJ
03B5 6502     226          ADC M
03B7 8506     227          STA INDICJ
03B9 60       228          RTS
03BA          229   ;
03BA          230   PATTRN:
03BA          231   ;
03BA          232   ; ------------------------------------------------
03BA          233   ;  ROOM FOR A "PATTERN" STRING UP TO 15 CHARACTERS
03BA          234   ; ------------------------------------------------
03C9          235          DFS 15
03C9          236   ;
03C9          237   STOP:
```

Figure 1

```
4   PRINT  TAB( 14);"**************"
5   PRINT  TAB( 14);"* EXAMPLE 1 *"
6   PRINT  TAB( 14);"**************"
10  PRINT  CHR$ (4)"BLOAD AMPER-POS.OBJ"
20  POKE 1014,0: POKE 1015,3: REM FETCH THE "&" AMPERSAND
30 A$ = "!":B$ = "THIS!IS!A!TEST"
40  & P(A$,B$,PO%)
50  GOSUB 100
60  CALL 880: GOSUB 100
70  IF PO% <  > 0 THEN 60
80  END
90  REM  PRINTING SUBROUTINE
100  IF PO% = 0 THEN  PRINT : PRINT "NO (OR NO MORE) OCCURENCE OF <";A$;"
     >": RETURN
110  PRINT : PRINT "THERE IS AN OCCURENCE OF <";A$;"> IN <";B$;"> AT THE
     ";PO%;"TH POSITION": RETURN
```

(Continued on next page)

Lines 200-216 put the result of the search in the output variable (zero if the search is unsuccessful).

Lines 221-225 prepare the INDICJ variable for the next searching.

See figure 1 for an example of how to use AMPER-POS.

If you save a name "SMITH", an address "31ST AV NY", and a phone number "333 444 555", on a mailing sequential file, you can use the sequence shown in figure 2.

Figure 2

```
10  PRINT "**************"
20  PRINT "* EXAMPLE 2 *"
30  PRINT "**************"
40  PRINT
50 D$ =  CHR$ (4)
60  PRINT D$"OPEN MAIL"
70  PRINT D$"WRITE MAIL"
80  PRINT "SMITH":

    PRINT "31ST AV N.Y."

    PRINT "333 444 5555"

90  PRINT D$"CLOSE MAIL"
```

But because it is better to have only one PRINT statement, we concatenate the three strings, separated with a "!". The statement 40 becomes:

40 A$ = "SMITH!31st AV NY!333 444 555!"

and statement 80 becomes:

80 PRINT A$

so that figure 3 will look like:

Figure 3

```
10  PRINT "**************"
20  PRINT "* EXAMPLE 3 *"
30  PRINT "**************"
40 A$ = "SMITH!31ST AV

        N.Y.!333  444 5555!"

50 D$ =  CHR$ (4)
60  PRINT D$"OPEN MAIL"
70  PRINT D$"WRITE MAIL"
80  PRINT A$
90  PRINT D$"CLOSE MAIL"
```

And to re-read the three values we use the program in figure 4.

References

1. *Microcomputer Problem Solving with Pascal*, Kenneth L. Bowles, Springer-Verlag, NY.

2. *BASIC Programming Reference Manual: Applesoft II*, Apple Computer Inc.

3. *Data Type and Structures*, C.C. Gotlieb, Leo R. Gotlieb, Prentice Hall Inc.

4. "A Comparison of Three String-Matching Algorithms," G. de V. Smit, *Software Practice and Experience*, Vol. 12, No. 1.

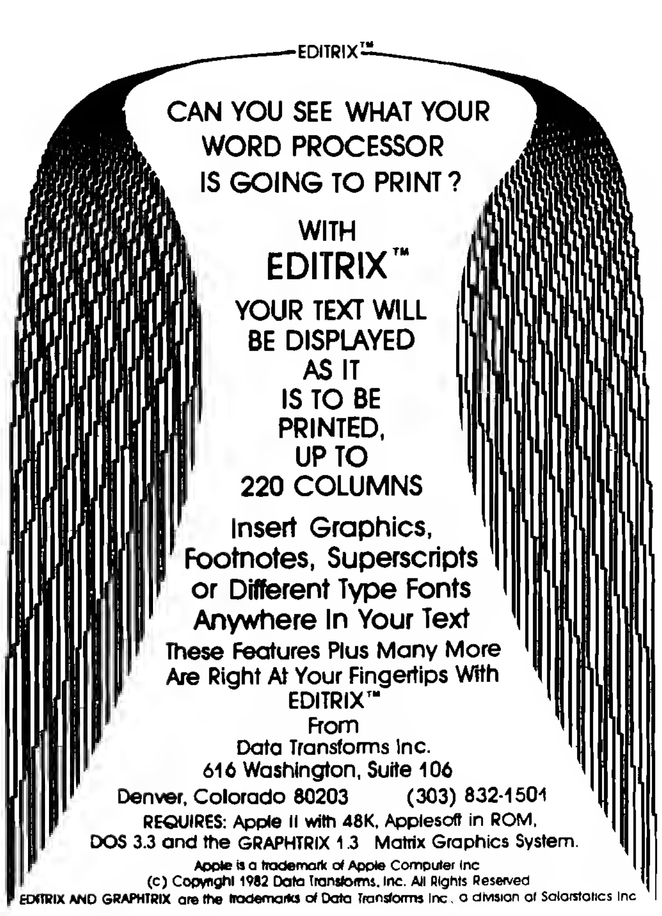
The author may be contacted at C.N.R.S./L.I.S.H., 31, Chemin Joseph Aiguier, 13277 Marseille Cedex 9, France.

Figure 1 *(Continued)*

```
THERE IS AN OCCURENCE OF <!> IN <THIS!IS!A!TEST> AT THE 5TH POSITION

THERE IS AN OCCURENCE OF <!> IN <THIS!IS!A!TEST> AT THE 8TH POSITION

THERE IS AN OCCURENCE OF <!> IN <THIS!IS!A!TEST> AT THE 10TH POSITION

NO (OR NO MORE) OCCURENCE OF <!>
```

Figure 4

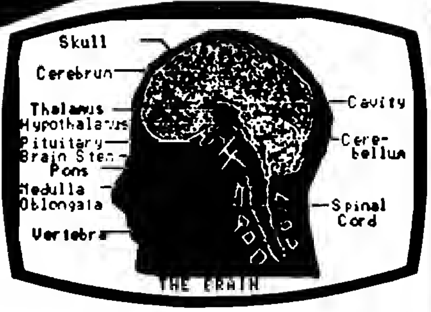
```
10 O$ =  CHR$ (4)
15  PRINT O$"BLOAD AMPER-POS.OBJ"
16  POKE 1014,0: POKE 1015,3: REM  LOAO & VECTOR
20  PRINT O$"OPEN MAIL"
30  PRINT D$"READ MAIL"
40  INPUT B$
50  PRINT D$"CLOSE MAIL"
60 A$ = "!"
70  & P(A$,B$,PO%)
80 NA$ =  MID$ (B$,1,PO% - 1):J = PO% + 1
90  CALL 880:AD$ =  MID$ (B$,J,PO% - J):J = PO% + 1
100  CALL 880:PN$ =  MID$ (B$,J,PO% - J)
```



Versa Computing

PRESENTS

VersaWriter DRAWING TABLET



COMPLETE HARDWARE / SOFTWARE GRAPHICS SYSTEM - $299

- Hi-Res & Med-Res Draw
- Paint Brush-5 Sizes
- Point to Point / Line Draw
- Air Brush
- Color Fill-In
- Change Color Hue & Intensity
- Reverse Picture
- Scaling
- Split / Full Screen
- Save / Load / Erase
- Text Writer
- Fix X or Y Axis

Requires: Atari ®300, 32K RAM, Basic Language Cartridge, Disk Drive

GRAPHICS COMPOSER

PADDLE / JOYSTICK GRAPHICS SOFTWARE - $39.95

- Draw on Hi-Res Screens 7 or 8
- Save Pictures on Disk or Cassette
- Create Player / Missile Shapes Automatically
- Geometric Figures Program
- Add Text to Screen

Requires: Atari®800, 32K RAM, Basic Language Cartridge, Disk or Cassette

GLOBE MASTER

COMPLETE HI-RES GEOGRAPHY GAME - $29.95

- 8 Hi-Res Color Maps
- U.S.-Europe-World-Asia-Africa-Australia
- Countries-Cities-Capitals-Oceans-Rivers-Mountains, Etc.
- Several Skill Levels

Requires: Atari®800, 32K RAM, Basic Language Cartridge, Disk

ATARI®is a registered trademark of Atari Inc.

MIND BOGGLERS

THREE STRATEGY GAMES

- Capture
- Mystery Box
- Simon Says

CASSETTE $15.95 DISK $19.95

Requires: Atari®400, 16K RAM, Cassette
Atari 800, 24K RAM, Cassette or Disk

Versa Computing, Inc.

3541 Old Conejo Road, Suite 104
Newbury Park, CA. 91320 **(805)498-1956**

OS-9 MATCH POWER WITH POWER 6809

INCREDIBLE

OS-9*** Tested and proven worldwide in industrial automation, communications, data processing and instrumentation.

16 bit performance 8 bit economy

OS-9. The modular, multitasking, Unix*-style operating system from Microware.

OS-9 from Microware unleashes the power of the amazing 6809 microprocessor.

Friendly.

Single or multi-user versatility.

"Pipes", I/O redirection and multi-level directories.

Modular, ROMable and easy to customize. Readily interfaces with almost any I/O device.

Supports up to 1 megabyte of memory.

No other system gives you all that and more.

Extensive Language Support

BASIC09*** — interactive structured Basic Compiler.

C Compiler — Unix version 7 compatible.

PASCAL — ISO standard native code compiler.

CIS COBOL** — ANSI 1974 standard COBOL.

With OS-9, software development in high level languages and debugging is faster and easier. Projects can be completed in a fraction of the time and cost.

Send for detailed literature.

There is much more you may want to know. See your computer supplier or contact us personally.

MICROWARE®

Microware Systems Corporation
5835 Grand Avenue
Des Moines, Iowa 50312
(515) 279-8844 Telex 910-520-2535

*Unix is a trademark of Bell Laboratories. **CIS COBOL is a trademark of Micro Focus, Ltd. ***OS-9 and BASIC09 are trademarks of Microware and Motorola, Inc.

6809 Macros for Structured Programming

by Hal Clark

This article presents a technique for using assembler macros to allow structured assembly-language programming. The examples presented are for the 6809, but they can be easily modified for other processors.

Structured Macros
requires:

6809-based machine with macro assembler
May be adapted to other processors

Structured program development is defined as *top down programming.* When using this approach, the main body of the program is developed first, then each succeeding level is developed until the total program is finished. When the program is entered into the computer, it is usually entered in the same, top down, fashion.

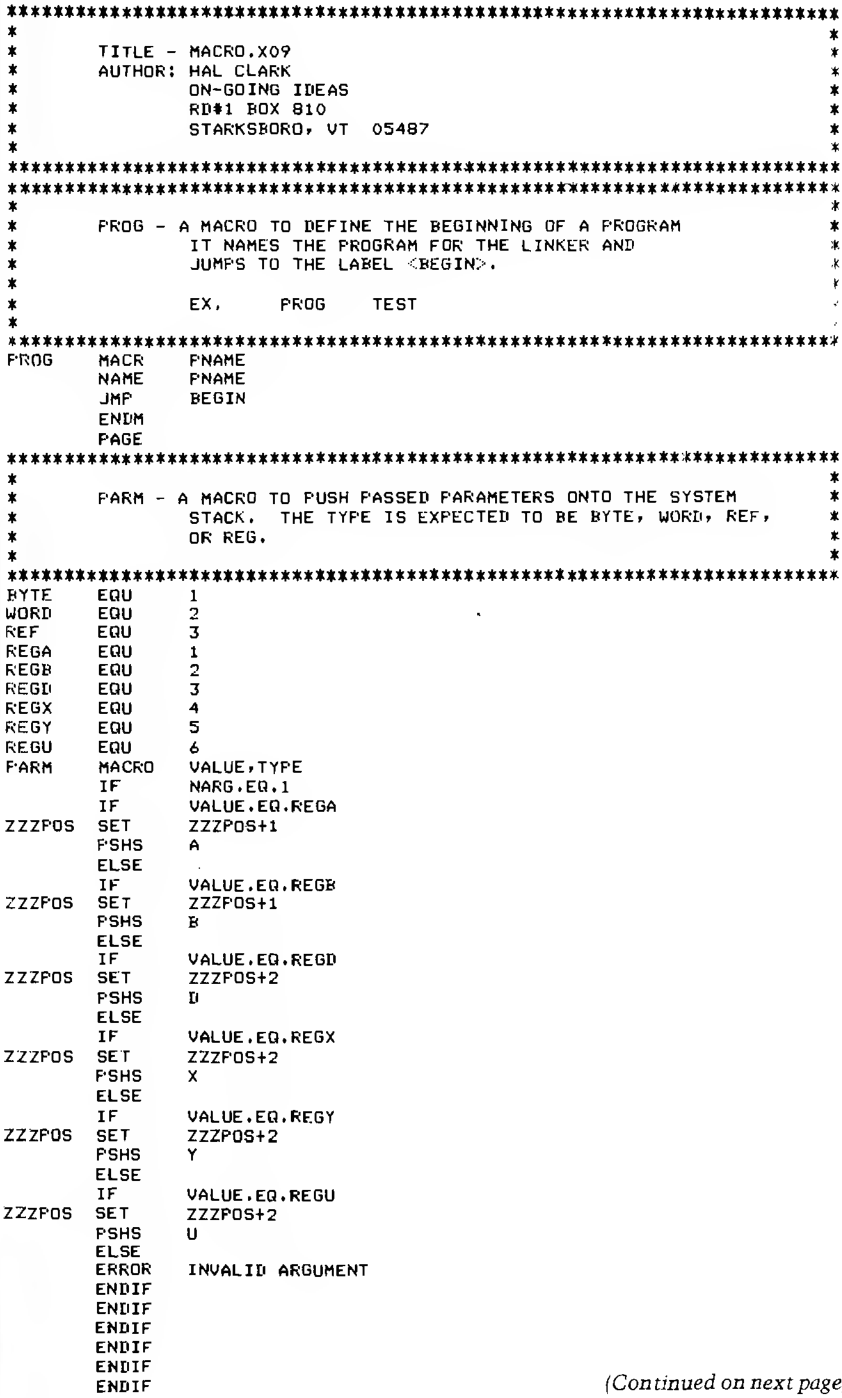
The newer structured languages, such as ADA, Pascal, and Modula, require that constants, variables, and procedures/functions (subroutines) be defined before use, thus giving the following program structure:

PROGRAM name
CONSTant definitions
VARiable definitions
FUNCTION definitions
PROCEDURE definitions
main program

BASIC and machine language, however, do not require that constants, variables, or subroutines be defined before use. Neither BASIC nor machine language requires program structuring. A lack of program structuring often results in an excessive use of memory and an unreasonable amount of time devoted to debugging, and customer support.

Pascal is a highly strucutred language that allows the systematic

Listing 1



```
*************************************************************************
*                                                                       *
*       TITLE - MACRO.X09                                               *
*       AUTHOR: HAL CLARK                                               *
*               ON-GOING IDEAS                                          *
*               RD#1 BOX 810                                            *
*               STARKSBORO, VT  05487                                   *
*                                                                       *
*************************************************************************
*************************************************************************
*                                                                       *
*       PROG - A MACRO TO DEFINE THE BEGINNING OF A PROGRAM             *
*               IT NAMES THE PROGRAM FOR THE LINKER AND                 *
*               JUMPS TO THE LABEL <BEGIN>.                             *
*                                                                       *
*               EX,     PROG    TEST                                    *
*                                                                       *
*************************************************************************
PROG    MACR    PNAME
        NAME    PNAME
        JMP     BEGIN
        ENDM
        PAGE
*************************************************************************
*                                                                       *
*       PARM - A MACRO TO PUSH PASSED PARAMETERS ONTO THE SYSTEM        *
*               STACK.  THE TYPE IS EXPECTED TO BE BYTE, WORD, REF,     *
*               OR REG.                                                 *
*                                                                       *
*************************************************************************
BYTE    EQU     1
WORD    EQU     2
REF     EQU     3
REGA    EQU     1
REGB    EQU     2
REGD    EQU     3
REGX    EQU     4
REGY    EQU     5
REGU    EQU     6
PARM    MACRO   VALUE,TYPE
        IF      NARG.EQ.1
        IF      VALUE.EQ.REGA
ZZZPOS  SET     ZZZPOS+1
        PSHS    A
        ELSE
        IF      VALUE.EQ.REGB
ZZZPOS  SET     ZZZPOS+1
        PSHS    B
        ELSE
        IF      VALUE.EQ.REGD
ZZZPOS  SET     ZZZPOS+2
        PSHS    D
        ELSE
        IF      VALUE.EQ.REGX
ZZZPOS  SET     ZZZPOS+2
        PSHS    X
        ELSE
        IF      VALUE.EQ.REGY
ZZZPOS  SET     ZZZPOS+2
        PSHS    Y
        ELSE
        IF      VALUE.EQ.REGU
ZZZPOS  SET     ZZZPOS+2
        PSHS    U
        ELSE
        ERROR   INVALID ARGUMENT
        ENDIF
        ENDIF
        ENDIF
        ENDIF
        ENDIF
        ENDIF
```

(Continued on next page)

development of computer programs. These programs are easy to debug and modify due to their structure. Unfortunately, not all applications can afford the overhead in memory usage and execution speed that many versions of Pascal require. The most transportable versions of Pascal compile the source code into some form of intermediate code, usually called P-code, which is then interpreted each time the program is run.

The macros presented in this article allow structured development of machine-language programs. Programs developed using these macros appear to be a combination of Pascal and in-line assembly language. By using macros, the structure of Pascal may be combined with the execution speed and memory efficiency of assembly language.

These macros generate machine code for the Motorola 6809 CPU. They are designed to be used with the Microtek 6809 cross-assembler, which is written in FORTRAN and available for most computers. The version used in the examples is running on a Digital Equipment Corp. PDP 11/34.

Although these macros generate instructions for a specific microprocessor, the ideas behind their design may be used to develop similar macros for other processors. Even if a macro assembler is not available, the ideas presented here can be used in machine-language programs to increase reliability and decrease development time.

Implementation

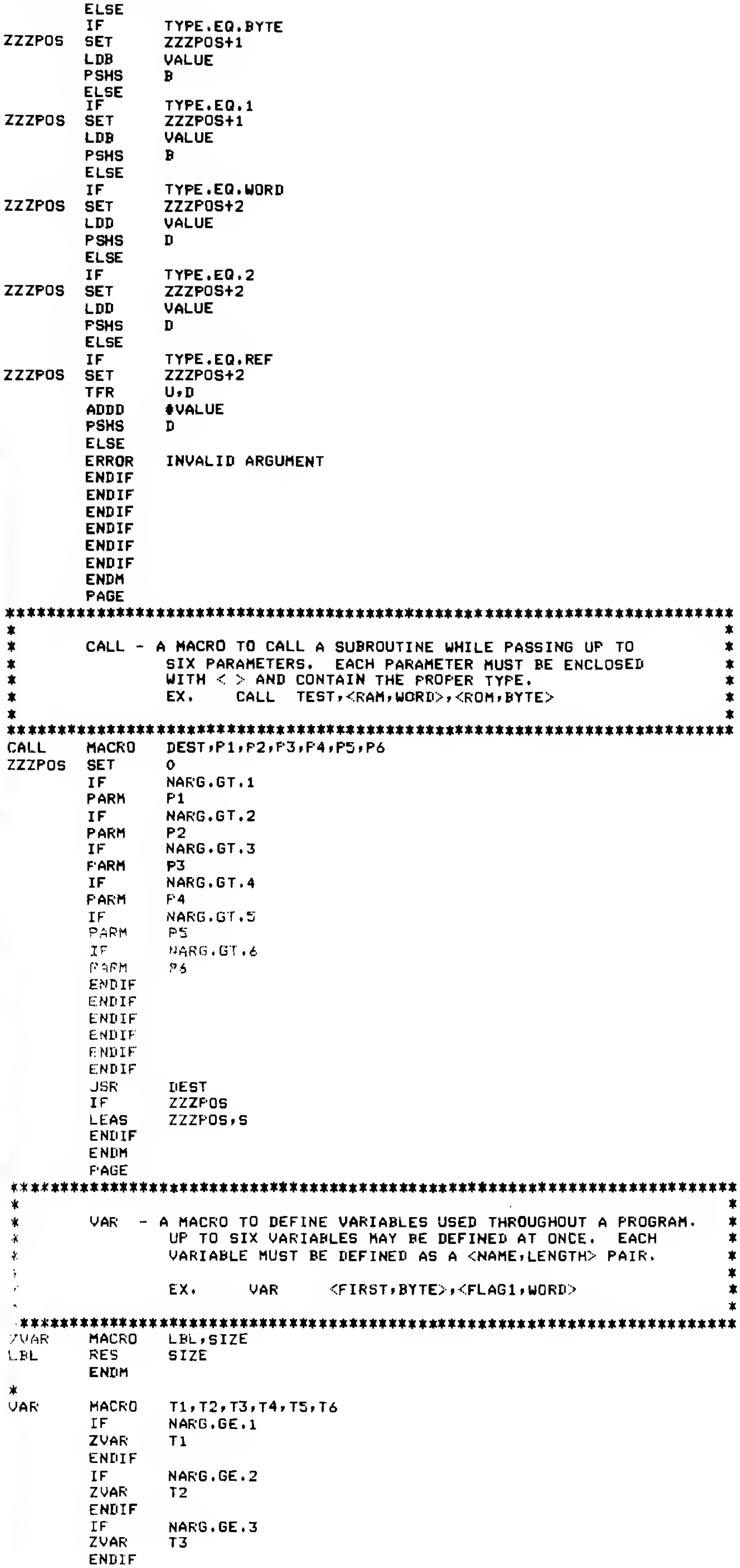
In this section we shall discuss the actual implementation of the macros, as well as some other aspects of structured programming not required by the use of the macros.

Macros, while allowing the program to have a structure similar to Pascal, do not actually require the same format (PROGRAM, CONSTANT, VARIABLE, FUNCTION, PROCEDURE, main program). However, most programs are easier to understand with that structure, so all the examples will use it.

The Stack Machine

These macros and programs written using the macros define the 6809-based computer as a stack-oriented machine. The 6809 makes this implementation easy because of its two stack registers and many addressing modes that allow stack access.

Listing 1 *(continued)*



```
        ELSE
        IF      TYPE.EQ.BYTE
ZZZPOS  SET     ZZZPOS+1
        LDB     VALUE
        PSHS    B
        ELSE
        IF      TYPE.EQ.1
ZZZPOS  SET     ZZZPOS+1
        LDB     VALUE
        PSHS    B
        ELSE
        IF      TYPE.EQ.WORD
ZZZPOS  SET     ZZZPOS+2
        LDD     VALUE
        PSHS    D
        ELSE
        IF      TYPE.EQ.2
ZZZPOS  SET     ZZZPOS+2
        LDD     VALUE
        PSHS    D
        ELSE
        IF      TYPE.EQ.REF
ZZZPOS  SET     ZZZPOS+2
        TFR     U,D
        ADDD    #VALUE
        PSHS    D
        ELSE
        ERROR   INVALID ARGUMENT
        ENDIF
        ENDIF
        ENDIF
        ENDIF
        ENDIF
        ENDIF
        ENDM
        PAGE
************************************************************************
*                                                                      *
*       CALL - A MACRO TO CALL A SUBROUTINE WHILE PASSING UP TO        *
*               SIX PARAMETERS.  EACH PARAMETER MUST BE ENCLOSED       *
*               WITH < > AND CONTAIN THE PROPER TYPE.                  *
*               EX.    CALL  TEST,<RAM,WORD>,<ROM,BYTE>                *
*                                                                      *
************************************************************************
CALL    MACRO   DEST,P1,P2,P3,P4,P5,P6
ZZZPOS  SET     0
        IF      NARG.GT.1
        PARM    P1
        IF      NARG.GT.2
        PARM    P2
        IF      NARG.GT.3
        PARM    P3
        IF      NARG.GT.4
        PARM    P4
        IF      NARG.GT.5
        PARM    P5
        IF      NARG.GT.6
        PARM    P6
        ENDIF
        ENDIF
        ENDIF
        ENDIF
        ENDIF
        ENDIF
        JSR     DEST
        IF      ZZZPOS
        LEAS    ZZZPOS,S
        ENDIF
        ENDM
        PAGE
************************************************************************
*                                                                      *
*       VAR  - A MACRO TO DEFINE VARIABLES USED THROUGHOUT A PROGRAM.  *
*               UP TO SIX VARIABLES MAY BE DEFINED AT ONCE.  EACH      *
*               VARIABLE MUST BE DEFINED AS A <NAME,LENGTH> PAIR.      *
*                                                                      *
*               EX.     VAR     <FIRST,BYTE>,<FLAG1,WORD>              *
*                                                                      *
************************************************************************
ZVAR    MACRO   LBL,SIZE
LBL     RES     SIZE
        ENDM
*
VAR     MACRO   T1,T2,T3,T4,T5,T6
        IF      NARG.GE.1
        ZVAR    T1
        ENDIF
        IF      NARG.GE.2
        ZVAR    T2
        ENDIF
        IF      NARG.GE.3
        ZVAR    T3
        ENDIF
```

Listing 1 *(continued)*

```
        IF      NARG.GE.4
        ZVAR    T4
        ENDIF
        IF      NARG.GE.5
        ZVAR    T5
        ENDIF
        IF      NARG.EQ.6
        ZVAR    T6
        ENDIF
        ENDM
        PAGE
************************************************************************
*                                                                      *
*       CONS  - A MACRO TO DEFINE PROGRAM CONSTANTS.  UP TO SIX        *
*               CONSTANTS MAY BE DEFINED AT ONCE USING THE STANDARD    *
*               <NAME,VAL> PAIR DEFINITIONS.                           *
*                                                                      *
*               EX.     CONS    <TRUE,#1>,<FALSE,#0>,<MAYBE,#3>        *
*                                                                      *
************************************************************************
ZCONS   MACRO   LBL,VAL
LBL     EQU     VAL
        ENDM
*
CONS    MACRO   T1,T2,T3,T4,T5,T6
        IF      NARG.GE.1
        ZCONS   T1
        ENDIF
        IF      NARG.GE.2
        ZCONS   T2
        ENDIF
        IF      NARG.GE.3
        ZCONS   T3
        ENDIF
        IF      NARG.GE.4
        ZCONS   T4
        ENDIF
        IF      NARG.GE.5
        ZCONS   T5
        ENDIF
        IF      NARG.EQ.6
        ZCONS   T6
        ENDIF
        ENDM
        PAGE
************************************************************************
*                                                                      *
*       LCONS - A MACRO TO DEFINE PROCEDURE CONSTANTS.  UP TO SIX      *
*               CONSTANTS MAY BE DEFINED AT ONCE USING THE STANDARD    *
*               <NAME,VAL> PAIR DEFINITIONS.  THESE CONSTANTS ARE      *
*               VALID ONLY FOR THE LIFE OF THE PROCEDURE.              *
*                                                                      *
*               EX.     LCONS   <TRUE,#1>,<FALSE,#0>,<MAYBE,#3>        *
*                                                                      *
************************************************************************
ZLCONS  MACRO   LBL,VAL
LBL     SET     VAL
        ENDM
*
LCONS   MACRO   T1,T2,T3,T4,T5,T6
        IF      NARG.GE.1
        ZLCONS  T1
        ENDIF
        IF      NARG.GE.2
        ZLCONS  T2
        ENDIF
        IF      NARG.GE.3
        ZLCONS  T3
        ENDIF
        IF      NARG.GE.4
        ZLCONS  T4
        ENDIF
        IF      NARG.GE.5
        ZLCONS  T5
        ENDIF
        IF      NARG.EQ.6
        ZLCONS  T6
        ENDIF
        ENDM
        PAGE
************************************************************************
*                                                                      *
*       PROC - A MACRO TO DEFINE PROCEDURE (SUBROUTINE) ENTRY POINTS   *
*               COMPLETE WITH UP TO SIX PARAMETERS.  NOTE IF DEFINED   *
*               WITH X PARAMETERS EACH CALL MUST USE X ALSO.           *
*                                                                      *
*          N O T E  -  1. THESE PARAMETERS ARE REFERENCED BY <PARM>,U  *
*                         IF THE PROCEDURE NEEDS TEMPS THEY SHOULD BE  *
*                         ALLOCATED USING THE LVAR MACRO.              *
*                                                                      *
*                      2. REGS X & Y ARE AUTOMATICALLY SAVED           *
************************************************************************
FPAR    MACRO   LBL,SIZE
```

(continued)

All data in the system, except for constants and variables defined in the main body of a program, are stored on the stack. Each procedure or function has the capability of receiving up to six passed parameters each time it is used. Procedures and functions also have a reserved area on the stack for their own use while active. When a procedure or function is finished, the area on the stack is given up so that the next subroutine may use it. This keeps the amount of "dedicated-all-the-time, used-some-of-the-time" memory to a minimum. Use of the stack also avoids the problems caused by using a common variable at the wrong time.

Register Usage

The Motorola 6809 has five 16-bit registers, two 8-bit registers, and one 16-bit accumulator that may be used as two 8-bit accumulators. The registers and their function within the defined stack machine follow:

16-bit

PC — program counter. Points to the next instruction to execute, and is only modified by the jump to subroutine, return from subroutine, and the various branch instructions.

S — system stack pointer. Must be initialized at the start of a program. This register should not be changed by the user program as it contains the necessary information to switch between subroutines.

U — user stack pointer. Data that is passed to a called subroutine may be found in this stack, and is referenced by an offset from this register. The value stored in U should never be changed by the user; the macros CALL and RETURN will alter U as necessary.

X — a general purpose register available to the user program. This register is saved and restored by the macros.

Y — a general purpose register available to the user program. This register is saved and restored by the macros.

8-bit

DP — direct page register. Unused by the stack machine.

CC — condition code register. This contains flags pertaining to the results of the last instruction. This register is not saved by the macros.

Accumulator D (A and B)

Accumulators A and B are each 8 bits. Accumulator D is 16 bits, consisting of the A and B accumulators. These registers are considered temporary and will be modified by the stack machine whenever a subroutine is called.

Reserved Symbols

The following symbols are restricted to reference only, and must not be redefined by the user's program:

BYTE used to represent a length of 8 bits.

WORD used to represent a length of 16 bits (2 bytes).

REF used when passing parameters between subroutines.

REGA used to pass the machine register A.

REGB used to pass the machine register B.

REGD used to pass the machine registers A and B.

REGX used to pass the machine register X.

REGY used to pass the machine register Y.

The Macros

The following paragraphs describe the available macros, their uses, calling parameters, and an example.

PROG xxxx

PROG: This macro expects one parameter that will name the program. The macro causes a jump to the symbol BEGIN, which should be placed on the first line of the main body of the program.

ex. PROG TEST

CALL xxx,<P1,TP>,...,<P6,TP>

CALL: Performs the jump to subroutine xxx after all the optional passed parameters are placed on the stack.

ex. CALL COPY,<FROM,WORD>,<TO,WORD>,<LEN,BYTE>

VAR <N1,TP>,...,<N6,TP>

VAR: Defines variables that will be used throughout the program, not just within subroutines.

ex. VAR <FLAG,BYTE>,<TEMP1,WORD>

CONS <N1,V1>,...,<N6,V6>

Listing 1 *(continued)*

```
LBL      SET      ZZZPOS
ZZZPOS   SET      ZZZPOS+SIZE
         ENDM
*
SAVE     EQU      1
*
PROC     MACRO    NA,F1,F2,F3,F4,F5,F6
NA       EQU      *                                PROCEDURE NAME
ZZZFNC   SET      0                                TELL RETURN NO PARMS TO PASS
ZZZENV   SET      0
ZZZSAV   SET      1
         IF       NARG.GT.1                        PASSING PARAMETERS ?
         PSHS     U,X,Y                                    YES - SAVE X,Y,U RESET S
         LEAU     4,S
ZZZENV   SET      1                                    SHOWTHAT THEY MUST BE CLEANED
ZZZPOS   SET      4                                        FROM THE STACK.
         IF       NARG.GT.2
         IF       NARG.GT.3
         IF       NARG.GT.4
         IF       NARG.GT.5
         IF       NARG.GT.6
         FPAR     F6
         ENDIF
         FPAR     F5
         ENDIF
         FPAR     F4
         ENDIF
         FPAR     F3
         ENDIF
         FPAR     F2
         ENDIF
         FPAR     F1
         ELSE                                    NO PARAMETERS PASSED - SAVE X,Y
         PSHS     X,Y
         ENDIF
ZZZPOS   SET      -4
         ENDM
         PAGE
************************************************************************
*      FPROC  - A MACRO TO DEFINE PROCEDURE (SUBROUTINE) ENTRY POINTS   *
*                COMPLETE WITH UP TO SIX PARAMETERS.  NOTE IF DEFINED   *
*                WITH X PARAMETERS EACH CALL MUST USE X ALSO.           *
*                                                                       *
*           N O T E  - 1. THESE PARAMETERS ARE REFERENCED BY <PARM>,U   *
*                         IF THE PROCEDURE NEEDS TEMPS THEY SHOULD BE   *
*                         ALLOCATED USING THE LVAR MACRO.               *
*                                                                       *
*                        2. REGS X & Y ARE NOT SAVED                    *
************************************************************************
FPROC    MACRO    NA,F1,F2,F3,F4,F5,F6
NA       EQU      *                                PROCEDURE NAME
ZZZFNC   SET      0                                TELL RETURN NO PARMS TO PASS
ZZZENV   SET      0
ZZZSAV   SET      0                                NOT SAVING REGS X,Y
         IF       NARG.GT.1                        PASSING PARMS?
         PSHS     U                                        YES - SAVE U
         LEAU     0,S                                      NO - DO NOTHING
ZZZENV   SET      1                                        THAT THEY MUST BE CLEANED
ZZZPOS   SET      4                                        FROM THE STACK.
         IF       NARG.GT.2
         IF       NARG.GT.3
         IF       NARG.GT.4
         IF       NARG.GT.5
         IF       NARG.GT.6
         FPAR     F6
         ENDIF
         FPAR     F5
         ENDIF
         FPAR     F4
         ENDIF
         FPAR     F3
         ENDIF
         FPAR     F2
         ENDIF
         FPAR     F1
         ENDIF
ZZZPOS   SET      0
         ENDM
         PAGE
************************************************************************
*        FUNC - A MACRO TO DEFINE FUNCTION  (SUBROUTINE) ENTRY POINTS   *
*               COMPLETE WITH UP TO SIX PARAMETERS.  NOTE IF DEFINED    *
*               WITH X PARAMETERS EACH CALL MUST USE X ALSO.            *
*               BY DEFINITION A FUNCTION RETURNS A SINGLE VALUE.        *
*                                                                       *
*          N O T E  - 1. THESE PARAMETERS ARE REFERENCED BY <PARM>,U    *
*                        IF THE FUNCTION  NEEDS TEMPS THEY SHOULD BE    *
*                        ALLOCATED USING THE LVAR MACRO.                *
*                                                                       *
*                       2. REGS X & Y ARE SAVED.                        *
************************************************************************
FUNC     MACRO    NA,F1,F2,F3,F4,F5,F6
NA       EQU      *                                FUNCTION  NAME
```

Listing 1 *(continued)*

```
ZZZFNC  SET     1                       TELL RETURN NO PARMS TO PASS
ZZZENV  SET     0
ZZZSAV  SET     1
        IF      NARG.GT.1               PASSING PARAMETERS ?
        PSHS    U,X,Y                           YES - SAVE X,Y,U RESET S
        LEAU    4,S
ZZZENV  SET     1                               THAT THEY MUST BE CLEANED
ZZZPOS  SET     4                               FROM THE STACK.
        IF      NARG.GT.2
        IF      NARG.GT.3
        IF      NARG.GT.4
        IF      NARG.GT.5
        IF      NARG.GT.6
        FPAR    F6
        ENDIF
        FPAR    F5
        ENDIF
        FPAR    F4
        ENDIF
        FPAR    F3
        ENDIF
        FPAR    F2
        ENDIF
        FPAR    F1
        ELSE                                    NO - SAVE X,Y
        PSHS    X,Y
        ENDIF
ZZZPOS  SET     -4
        ENDM
        PAGE

************************************************************************
*                                                                      *
*     FFUNC  - A MACRO TO DEFINE FUNCTION  (SUBROUTINE) ENTRY POINTS   *
*               COMPLETE WITH UP TO SIX PARAMETERS.  NOTE IF DEFINED   *
*               WITH X PARAMETERS EACH CALL MUST USE X ALSO.           *
*               BY DEFINITION A FUNCTION RETURNS A VALUE.              *
*                                                                      *
*          N O T E  - 1.  THESE PARAMETERS ARE REFERENCED BY <PARM>,U  *
*                       IF THE FUNCTION  NEEDS TEMPS THEY SHOULD BE    *
*                       ALLOCATED USING THE LVAR MACRO.                *
*                                                                      *
*                     2.  REGS X & Y ARE NOT SAVED                     *
*
***********************************************************************-
FFUNC   MACRO   NA,F1,F2,F3,F4,F5,F6
NA      EQU     *                       FUNCTION  NAME
ZZZFNC  SET     1                       TELL RETURN NO PARMS TO PASS
ZZZENV  SET     0
ZZZSAV  SET     0                       NOT SAVING REGS X,Y
        IF      NARG.GT.1               PASSING PARMS?
        PSHS    U                               YES - SAVE U
        LEAU    0,S                             NO - DO NOTHING
ZZZENV  SET     1                               THAT THEY MUST BE CLEANED
ZZZPOS  SET     4                               FROM THE STACK.
        IF      NARG.GT.2
        IF      NARG.GT.3
        IF      NARG.GT.4
        IF      NARG.GT.5
        IF      NARG.GT.6
        FPAR    F6
        ENDIF
        FPAR    F5
        ENDIF
        FPAR    F4
        ENDIF
        FPAR    F3
        ENDIF
        FPAR    F2
        ENDIF
        FPAR    F1
        ENDIF
ZZZPOS  SET     0
        ENDM
        PAGE

************************************************************************
*                                                                      *
*       LVAR  - A MACRO TO DEFINE TEMPORARY VARIABLES FOR A PARTICULAR *
*               PROCEDURE OR FUNCTION.  THE TEMPS ARE ALLOCATED ON THE *
*               SYSTEM STACK AND ARE REFERENCED BY <TEMP>,U. THESE     *
*               VARIABLES ARE DISTROYED UPON EXIT OF THE PROCEDURE.    *
*                                                                      *
*               EX.                                                    *
*                       LVAR    <T1,BYTE>,<T2,WORD>                    *
*                                                                      *
*                       LDD     T2,U                                   *
*                                                                      *
************************************************************************
DPAR    MACRO   LBL,SIZE
ZZZPOS  SET     ZZZPOS-SIZE
LBL     SET     ZZZPOS
        ENDM
```

(continued)

CONS: Defines constants that will be used throughout the program.

ex. CONS <TRUE,1>,<FALSE,0>, <MAYBE,3>

LCONS <N1,V1>,...,<N6,V6>

LCONS: Defines local constants that are used within the current subroutine definition only.

ex. LCONS <TAB,9>,<CRLF,13>

PROC xxx,<P1,TP>,...,<P6,TP>

PROC: Defines a procedure (subroutine). PROC must be followed by a RETURN before the next PROC or FUNC macro usage. xxx gets assigned to the entry point of the procedure. The parameters P1-P6 get assigned as offsets on the stack to allow referencing of the passed parameters at execution.

ex. PROC COPY,<IN,WORD>, <OUT,WORD>,<LEN,BYTE>

FUNC xxx,<P1,TP>,...,<P6,TP>

FUNC: Defines a subroutine that returns a value (function). This macro, like PROC, must be followed by the RETURN macro before the next FUNC or PROC macro is used. The name xxx is assigned to the entry point of this subroutine and the labels P1-P6 are assigned as offsets to the stack for access to the passed data during execution. The results of the function are returned in machine register D.

ex. FUNC SQR,<VAL,WORD>

LVAR <N1,TP>,...,<N6,TP>

LVAR: Defines up to six local variables at one time. These variables can only be referenced within the definition of this defining subroutine. Subroutines called from within defining subroutines may reference these variables only if passed to the subroutine as a call parameter. LVAR, at this time, requires parameters to be passed to the defining subroutine to function properly.

ex. LVAR <TEMP1,BYTE>, <TEMP2,WORD>

RETURN data

RETURN: Ends the previous FUNCtion or PROCedure definition. It removes temporary variables as well as passed parameters from the stack, restores the saved registers, and returns to the calling program. If the RETURN is from a function, it also places the returned value in the machine register D.

ex. RETURN RESLT (from a function)
ex. RETURN (from a procedure)

The author may be contacted at On-Going Ideas, RD #1, Box 810, Starksboro, Vermont 05487.

Listing 1 *(continued)*

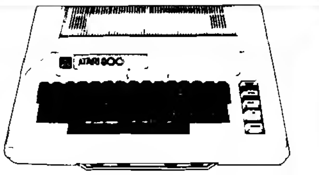
```
*
LVAR     MACRO    V1,V2,V3,V4,V5,V6
         IF       ZZZENV.EQ.0
ZZZENV   SET      -1
         PSHS     U
         LEAU     0,S
ZZZPOS   SET      0
         ENDIF
         IF       NARG.GE.1
         DPAR     V1
         IF       NARG.GE.2
         DPAR     V2
         IF       NARG.GE.3
         DPAR     V3
         IF       NARG.GE.4
         DPAR     V4
         IF       NARG.GE.5
         DPAR     V5
         IF       NARG.GE.6
         DPAR     V6
         ENDIF
         ENDIF
         ENDIF
         ENDIF
         ENDIF
         ENDIF
         LEAS     ZZZPOS,U
         ENDM
         PAGE


************************************************************************
*                                                                      *
*        RETURN - A MACRO TO DEALLOCATE PASSED PARAMETER STACK SPACE   *
*                 UPON THE COMPLETION OF A FUNCTION OR PROCEDURE.      *
*                                                                      *
*                 N O T E - FUNCTION WILL RETURN A VALUE IN REGA OR REGD *
*                         THIS VALUE IS SPECIFIED IN THE RETURN CALL.  *
*                                                                      *
*                         EX.     RETURN   <T1,U>,WORD                 *
*                                                                      *
************************************************************************
RETURN   MACRO    VAL,TYPE
         IF       ZZZFNC.EQ.1                  FUNCTION RETURN?
         IF       NARG.EQ.0                    YES - BUT NO PARMS(ERROR)
         ERROR    FUNCTION MUST RETURN A VALUE!!!!!!
         ELSE
         IF       NARG.EQ.1                    YES - MAYBE A REG IS RETURNED
         IF       VAL.EQ.REGA                            REGA ?
*                                                        A=A
         ELSE
         IF       VAL.EQ.REGD                            REGD ?
*                                                        D=D
         ELSE
         IF       VAL.EQ.REGB                            REGB ?
         TFR      B,A                                    A=B
         ELSE
         IF       VAL.EQ.REGX                            REGX ?
         TFR      X,D                                    D=X
         ELSE
         IF       VAL.EQ.REGY                            REGY ?
         TFR      Y,D                                    D=Y
         ELSE
         ERROR    INVALID RETURN ARGUMENT
         ENDIF
         ENDIF
         ENDIF
         ENDIF
         ENDIF
         ELSE
         IF       NARG.NE.2                              MORE THAN 1 RETURN ARG?
         ERROR    INVALID RETURN ARGUMENT
         ELSE
         IF       TYPE.EQ.BYTE                           RETURNING A BYTE?
         LDA      VAL                                            A=VAL
         ELSE
         IF       TYPE.EQ.WORD                           RETURNING A WORD?
         LDD      VAL                                            D=VAL
         ELSE
         ERROR    INVALID RETURN ARGUMENT
         ENDIF
         ENDIF
         ENDIF
         ENDIF
         ENDIF
         ELSE
         IF       NARG.NE.0
         ERROR    PROCEDURES DO NOT RETURN VALUES!!!
         ENDIF
         ENDIF
*
*        END OF SPECIAL FUNCTION RETURN LOGIC
*
```

Listing 1 *(continued)*

```
ZZZFNC  SET     0
        IF      ZZZENV.LT.0
        LEAS    0,U
        PULS    U
        IF      ZZZSAV
        PULS    X,Y
        ENDIF
        ELSE
        IF      ZZZENV.GT.0
        IF      ZZZSAV
        LEAS    -4,U
        PULS    U,X,Y
        ELSE
        LEAS    0,U
        PULS    U
        ENDIF
        ELSE
        IF      ZZZSAV
        PULS    X,Y
        ENDIF
        ENDIF
        ENDIF
        RTS
        ENDM
        PAGE
************************************************************************
*                                                                      *
*       RECORD DEFINITION MACROS                                       *
*                                                                      *
************************************************************************
RECORD  MACRO
ZZZPOS  SET     0
        ENDM
FIELD   MACRO   NAME,SIZE
NAME    EQU     ZZZPOS
ZZZPOS  SET     ZZZPOS+SIZE
        ENDM
RECSIZ  MACRO   SIZE
SIZE    EQU     ZZZPOS
        ENDM
        PAGE

************************************************************************
*                                                                      *
*       TABLE GENERATION MACROS                                        *
*               EACH TABLE MUST CONSIST OF:                            *
*                       TABLE - TABLE HEADER (1)                       *
*                       TABDTA - TABLE DATA ENTRIES (N)                *
*                       TABSIZ - TABLE LENGTH AND END MARKER (1)       *
*                                                                      *
************************************************************************
TABLE   MACR
ZZZTBL  SET     0
        ENDM
*
TABDTA  MACR    VALUE,TYPE
        IF      TYPE.EQ.BYTE
        DATA    VALUE
ZZZTBL  SET     ZZZTBL+1
        ELSE
        IF      TYPE.EQ.WORD
        ACON    VALUE
ZZZTBL  SET     ZZZTBL+2
        ELSE
        ERROR   UNSPECIFIED DATA TYPE
        ENDIF
        ENDIF
        ENDM
*
TABSIZ  MACR    SIZE
SIZE    EQU     ZZZTBL
        DATA    $FF
        ENDM
        PAGE
************************************************************************
*                                                                      *
*       STORE A STRING (MUST BE USED WITHIN A PSCT, OUTSIDE EXECUTABLE CODE)
*       THE PARAMETER IS THE STRING ENCLOSED IN SINGLE QUOTES, AS WELL AS
*       ANGLE BRACKETS, IF NECESSARY.                                  *
*                                                                      *
************************************************************************
MAKSTR  MACRO   XS
ZZZSVS  SET     *
        ORG     *+1
        FCC     XS
ZZZSTR  SET     *
        ORG     ZZZSVS
        FCB     ZZZSTR-ZZZSVS-1
        ORG     ZZZSTR
        ENDM
        PAGE
```

computer mail order west

ATARI



400

16K ... $269
32K ... $389
48K ... $489

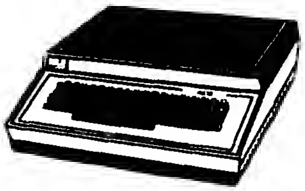
800

16K ... $649
32K ... $724
48K ... $769

410 Recorder	$76.00
810 Disc Drive	$449.00
822 Printer	$269.00
825 Printer	$589.00
830 Modem	$159.00
820 Printer	$259.00
850 Interface	$169.00
CX40 Joy Stick	$18.00
CX853 16K RAM	$77.95

Microtek 16K RAM	$74.95
Microtek 32K RAM	$119.95
Ramdisk (128K)	$429.95
Intec 48K Board	$219.95
Intec 32K	$119.95
One year extended warranty	$70.00
481 Entertainer	$69.00
482 Educator	$130.00
483 Programmer	$49.00
484 Communicator	$344.00

FRANKLIN ACE 1000



64K Personal Computer

Hardware, software and peripheral compatable with the Apple II and even has some features not found on the Apple.

Call now for best price!!

HOT ATARI GAMES

PAC-MAN	$35.00
Centipede	$35.00
Caverns of Mars	$32.00
Asteroids	$29.00
Missile Command	$29.00
Star Raiders	$39.00
Canyon Climber	$25.00
Protector	$24.00
Mouskattack	$31.00
Jawbreaker	$27.00
Ghost Hunter	$24.00

K-BYTE
ROM CARTRIDGE GAMES FOR YOUR ATARI

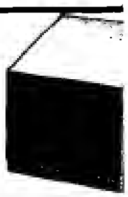
Krazy Shoot Out	$39.00
K-razy Kritters	$39.00
K-razy Antics	$39.00
K-star Patrol	$39.00

STICK STAND
$6.99

ARCADE ACTION FROM YOUR ATARI JOYSTICK

PERCOM

Finally, a dual disk drive for the Atari 800/400. Both single and dual drive models read both sides of the disk! Will read all disks written for Atari 810. *CALL FOR INTRODUCTORY PRICE!*



Telecommunications Modems

Hayes	
Smart	$239.00
Chronograph	$199.00
Micromodem II	$279.00
Micromodem 100	$309.00
Novation Auto	$239.00
D Cat	$169.00
Cat	$159.00
Anchor Modem	$79.00

AMDEK MONITORS

300G	$169.00
Color I	$339.00
Color II	$699.00
Color III	$429.00

OTHERS

Zenith 9" (Green)	$119.00
BMC 12" Green	$85.00

NEC

COMPUTERS

8001-A	$749.00
8031	$749.00
8012	$549.00

PRINTERS

8023	$549.00
7710/7730	$2399.00
3510/3530	$1789.00

MONITORS

JB-1201	$179.00
JC-1201	$349.00
JC-1202	$899.00

VISICORP

VISICALC	
Apple II +	$189.00
Atari	$189.00
Commodore	$189.00
IBM	$189.00

Also available are:

VISIDEX	VISIPLOT
VISIFILE	VISITERM
VISIPACK	VISITREND

SOFTWARE

We stock manufacturer's and third party software for most all computers on the market! Call today for a copy of our new

CATALOG

You'll find programs by Atari, APX, Data Soft Crystal Software, EPYX, Synapse, OnLine, Arcade Plus, K-BYTE, Magis, Canadian Micro Professional Software, Creative Software, BPI, VISICORP, Commodore, NEC and of course our own!

Maxell Disks

MD I (box of 10)	$36.(
MD II (box of 10)	$46.(
MFD I (8")	$44.(
MFD II (8" Double Density)	$54.(
Syncom (box of 10)	$29.(

Computer Covers

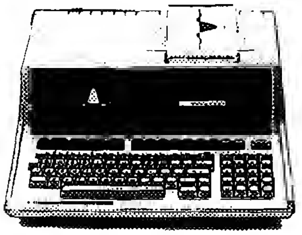
		Commodore VIC-20	$6.99
Atari 400	$6.99	Commodore 8032	$14.99
Atari 800	$6.99	Commodore	
Atari 810	$6.99	8050/4040	$10.99

west 800-648-3311

In Nevada
CALL
(702)588-5654

P.O. Box 66&
Statelin
NV. 894(

HEWLETT PACKARD



HP•85 $1969

HP•87	$1799.00
HP•125	$1999.00
HP•85 16K Memory Module	$169.00
5¼" Dual Master Disc Drive	$1799.00
Hard Disk w/ Floppy	$4349.00
Hard Disk	$3549.00
"Sweet Lips" Plotter	$1199.00
80 Column Printer	$649.00

CBM 8032 $999

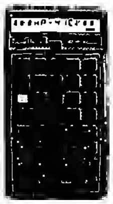
Word Pro 5 +	$319.00
Word Pro 4 +	$299.00
Word Pro 3 +	$199.00
The Administrator	$379.00
InfoPro Plus	$219.00
Power	$79.00

Commodore Business Machines

CBM 64	CALL
4032	$969.00
8096 Upgrade Kit	$369.00
Super Pet	$1599.00
2031	$529.00
8250 Doubled Sided Disk Drive	$1699.00
D9060 5 Megabyte Hard Disk	$2399.00
D9090 7.5 Megabyte Hard Disk	$2699.00
8050	$1299.00
4040	$969.00
8300 (Letter Quality)	$1799.00
8023	$769.00
4022	$499.00
Pet to IEEE Cable	$37.00
IEEE to IEEE Cable	$46.00
Tractor Feed for 8300	$240.00

HEWLETT PACKARD

HP 41CV CALCULATOR $239



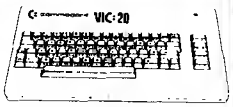
HP 41C	$189.00
HP 11C	$79.00
HP 12C	$114.00
NEW 15C	$119.00
NEW 16C	$125.00

HPIL PERIPHERALS IN STOCK!

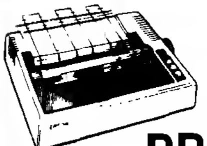
Televideo Terminals

910	$579
912C	$699
920C	$749
925C	$749
960	$939
802	$Call
802H	$Call
816	$Call
806	$Call

VIC 20 $239



VIC 1530 Commodore Datassette	$69.00
VIC 1540 Disk Drive	$499.00
VIC 1515 VIC Graphic Printer	$339.00
VIC 1210 3K Memory Expander	$32.00
VIC 110 8K Memory Expander	$53.00
16K VIC Expansion	$94.00
VIC 1011 RS232C Terminal Interface	$43.00
VIC 112 VIC IEEE-488 Interface	$86.00
VIC 1211 VIC 20 Super Expander	$53.00
VIC Mother Board	$99.00



EPSON PRINTERS

MX 80 w/Graftrax	$449
MX 80 FT III	CALL
MX 100	CALL
ADA 1600 Parallel Printer to CBM	$119.00
ATC-1 Parallel Printer to Atari	$29.00
AP-80 Apple Parallel Card & Cable	$69.00
IBM-1 Parallel Printer to IBM	$32.00

CITOH PRINTERS

Starwriter	
F10-40CPS	$1439.00
F10-55CPS	CALL
Prowriter	$499.00
ADA 1450 Serial Printer to CBM	$139.00
ATC-2 Serial Printer to Atari	$29.00
AP-S10 Apple Serial Card & Cable	$95.00

Smith-Corona

TP-I

$649

Letter Quality
Daisy Wheel
12 CPS Text Printer

In-stock items shipped same day you call. No risk, no deposit on C.O.D. orders. Pre-paid orders receive free shipping within the continental United States with no waiting period for certified checks or money orders. All prices shown are cash prices. Add 3% for Mastercard and Visa. NV and PA residents add sales tax. All items subject to availability and price change.

CENTRONICS PRINTERS

739-1	$519
739-3	$619
2 Meter RS232-RS232 Cables	$29.95

— ALSO —

Diablo 630 Special	$1799.00
Talley 8024-L	$1629.00
IDS Prism	CALL

OKIDATA PRINTERS

82A	$489.00
83A	$729.00
84	$1099.00

NOTE
All of the above OKIDATA Printers come equiped with both parallel ports & RS232 Serial ports.

800-233-8950 east

477 E.
THIRD ST.
Williamsport
PA 17701

IN PA
CALL
(717)327 9575

computer mail order east

Market Projection Program for the Color Computer

by Leonard I. Suckle

This sophisticated business program is implemented on an inexpensive Color Computer system. It may be modified for other computers.

Market Projection requires:

TRS-80 Color Computer with 16K
It does *not* require Extended Color BASIC

The introduction of low-cost personal computers has placed the capability of huge computational powers within the economic reach of nearly everyone. Unfortunately, many of the lower-priced systems are being promoted or received as either toys or games.

The program described in this article is an example of how the Radio Shack Color Computer may be used in an actual business application. The computer used here is the 16K version without the Extended BASIC ROM. It may be purchased for under $500. The addition of a line printer and a cassette tape recorder add approximately $300 to provide for a complete, versatile system.

The program is written in BASIC and may be used on any compatible system. Variations could easily be made to customize it to the users' specific needs. These applications should substantiate that the use of low-cost personal computers in business should not be overlooked.

Program Description

Making projections on industry and company sales, when a multitude of product lines are involved, can be a long and tedious task. A single change in value usually results in complete recalculation of the other parameters for each year of the projection. Knowledge of the required changes causes either a conscious or subconcious hesitation in the modification of data to provide more realistic interpretation of the market conditions of the future.

The intent of this computer program is to alleviate the tedious calculation and recalculation associated with revisions. With the capability of making changes and seeing their immediate effect on the rest of the data, more accurate decisions may be made in industry planning functions.

The market projection computer program evaluates a market and share of the market by year and by each of many product categories. The market parameters evaluated for each of the product categories are shown in table 1. The computer refers to each of these parameters by their item # (1 through 9) shown in the table and in figure 1 in brackets.

The total of all columns of data will be calculated upon request. Furthermore, a top-down summation value may be entered for TAM and NSB, and the difference between the actual and top-down summations will be displayed. Upon request, the computer will adjust the TAMs and NSBs, based on product mix, so they equal the top-down value.

Entering and working with two years of data at a time allows you to make projections and/or calculate compound growth rates. Data from one year may be projected by annual compound growth rate (%) to the other year in memory. Select any of the nine parameters for projection. Calculations will be done either on a product category basis or for all product categories. If data is entered for both years, the computer can calculate the growth rate of any data on an annual compound percentage.

***Table 1:* Market Parameters**

1. Company Market Share (NSB/TAM) in %
2. Total Available Market (TAM) in millions of $
3. Industry Product Mix (TAM of specific product/Total TAM) in %
4. Industry Quantity in millions of units
5. Industry Average Selling Price (ASP) in $
6. Company Sales (NSB) in millions of $
7. Company Product Mix (NSB of specific product/Total NSB) in %
8. Company Quantity in millions of units
9. Company Average Selling Price (ASP) in $

Data may be saved and loaded by year from cassette. Two years of data may be stored in memory for entry and calculation. Within any one of the two years, data may be calculated based on other data entered. This may be done on a product category basis, or for all product categories. For example, if the units and ASP for industry have been entered, the program will calculate the TAM. For expediency of operation, calculations are made only when requested, rather than whenever new data is entered.

Video Display Characteristics

Data for each of the two years stored in memory is presented in an array of pages for each year on the TV display of the Color Computer. These pages are arranged in a 3 × N matrix, where N is dependent upon the number of product categories.

Three vertical pages can display a maximum of 29 product categories. This example is shown in figure 1. The product categories are numbered and

16K RAM EXPANSION BOARD FOR THE APPLE II* $120.00

The Andromeda 16K RAM Expansion Board allows your Apple to use RAM memory in place of the BASIC Language ROMs giving you up to 64K of programmable memory. Separate Applesoft* or Integer BASIC ROM cards are no longer needed. The 16K RAM Expansion Board works with the Microsoft Z-80 card, Visicalc, DOS 3-3, Pascal, Fortran, Pilot, and other software. A switch on the card selects either the RAM language or the mainboard ROMs when you reset your Apple.

The Andromeda 16K RAM Expansion Board has a proven record for reliability with thousands of satisfied customers.

Now with One Year Warranty.

ANDROMEDA
INCORPORATED
Greensboro, NC. 27410
P.O. Box 19144
919 852-1482

VISA
master charge

Price for Andromeda 16K RAM expansion board now only $120.00. Please add $5 for shipping and handling. North Carolina residents add 4% sales tax.

*DEALER INQUIRIES WELCOME.

listed along the left side of the display, and the market parameters and their item numbers are displayed across the top.

The two years stored in memory are defined as the primary and secondary years, and are located on pages 1 through 3, and 4 through 6, respectively. When in the display mode of the program, the page being displayed may be changed by direct numerical entry of the desired page, or by indexing left, right, up, and down (see DISPLAY).

Printer Output

Data from either the primary or secondary year may be output to a printer connected to the Color Computer. The Quick Printer II interfaces directly, and provides a 32-character line width. The program prints the data within the year last displayed on the video. All pages of the displayed year will be printed independent of the page shown on the video display.

Operation

The program is loaded from cassette tape into memory using the CLOAD command. Sufficient memory must be allocated to the program (13,700 bytes). Although the program was written with standard ROM, it will also operate with an extended ROM computer, but a comparable amount of memory must be freed to permit operation. Extended BASIC will not free more memory than that amount created by the PCLEAR 1 command. This is not enough memory for this program, so an additional page of memory must be freed prior to loading. This may be done by entering the following command:

```
POKE 25,6 (enter)
```

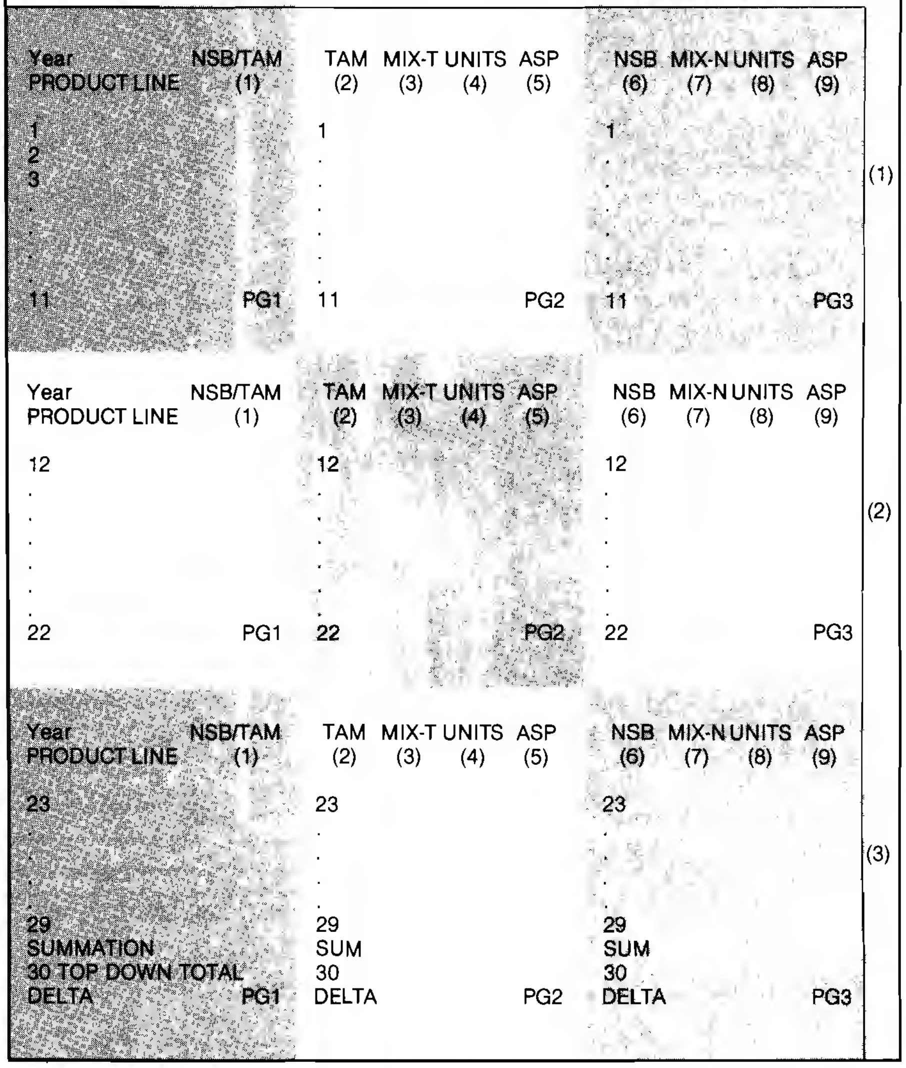
After the program is loaded, entering RUN initiates operation. The screen will clear and the program enters MAIN CONTROL. The program is initiated with all data cleared, primary year = 1981, secondary year = 1986, and the display page pointer set at 1,(1).

MAIN CONTROL

MAIN CONTROL is the principal operating routine from which all other routines may be activated. Operation is indicated by an inverted-screen (dark background) "M" in the lower left of the display, preceding the cursor.

Commands available from MAIN CONTROL are shown in table 2. The "enter" key must be used when operating in this routine.

Figure 1: Video Display Pages 1 through 3 for Primary Year (Secondary Year Pages are 4 through 6). Vertical Page is specified by "shift (n)", where n = 1 to 3. Item #'s are shown in brackets and are described in table 1.



```
Year             NSB/TAM   TAM  MIX-T UNITS  ASP    NSB  MIX-N UNITS  ASP
PRODUCT LINE       (1)     (2)   (3)   (4)   (5)    (6)   (7)   (8)   (9)

1                          1                        1
2                          .                        .
3                          .                        .                      (1)
.                          .                        .
.                          .                        .
.                          .                        .
11                 PG1     11                 PG2   11                PG3

Year             NSB/TAM   TAM  MIX-T UNITS  ASP    NSB  MIX-N UNITS  ASP
PRODUCT LINE       (1)     (2)   (3)   (4)   (5)    (6)   (7)   (8)   (9)

12                         12                       12
.                          .                        .
.                          .                        .                      (2)
.                          .                        .
.                          .                        .
22                 PG1     22                 PG2   22                PG3

Year             NSB/TAM   TAM  MIX-T UNITS  ASP    NSB  MIX-N UNITS  ASP
PRODUCT LINE       (1)     (2)   (3)   (4)   (5)    (6)   (7)   (8)   (9)

23                         23                       23
.                          .                        .
.                          .                        .                      (3)
.                          .                        .
29                         29                       29
SUMMATION                  SUM                      SUM
30 TOP DOWN TOTAL          30                       30
DELTA              PG1     DELTA              PG2   DELTA             PG3
```

DISPLAY

The display routine outputs the data for the primary or secondary years stored in memory to the video display. Operation in this routine is indicated by an inverted-screen "D" in the lower left of the display.

As previously described under Video Display Characteristics, each year is displayed in a 3 × N matrix of pages. Any page may be immediately displayed by entering the horizontal page number directly, and the vertical page number using the shift key. The display may also be changed by using the arrow of the appropriate direction (up, down, left, right). The "enter" key is not needed with entries from DISPLAY, and all actions will occur immediately upon depression of the appropriate key.

Commands available from DISPLAY are shown in table 3.

ENTRY

The entry routine permits entry of data into the designated column of the year last displayed on the video. An inverted-screen "E" is displayed in the lower left of the video screen to indicate operation in this routine. The "enter" key is required for entries in this routine.

Upon entering this routine, the display will prompt the user with the request:

ROW(* OR YEAR), ITEM #(*), VALUE (OR 'YR',,YEAR)

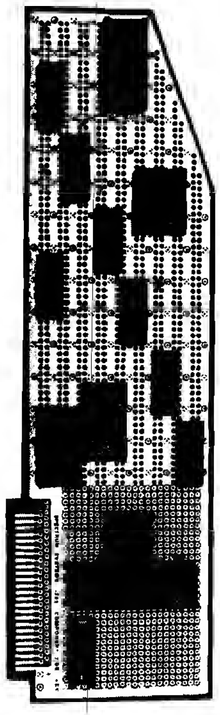
(Continued on page 71)

APPLE II PERIPHERAL DEVELOPERS:

Your complex function prototype requires
the best wirewrap board available.

SPECTRUM SYSTEMS .... MAKES IT!

Fully Extended Wirewrap Protoboard.



Size: 2.8 by 10.7 inch 2 layer PC.

Capacity: up to 58-16 pin or 12-40 pin
or any combination sockets inbetween.

Carefully designed +5 and GND planes provide for the minimum electrical noise, low impedance, hi capacitance, and maximum versatility in the layout of IC's, capacitors, discretes and I/O connectors.

Wire-wrap technique documentation included.

Terms:

- $45.00 + (6% Cal. Res. tax) + $2.00 S&H.
- All payments must be in U.S. funds drawn on a U.S. bank.
- Outside U.S. add 10%.
- Cashier check/money order allow 30 day ARO.
- Personal checks add 2 weeks.
- No credit cards or cash, Please!

Spectrum Systems
P.O. Box 2262
Santa Barbara, Ca. 93120

Apple II is a trademark of Apple Computers

M.P. Computer Services Corporation

2396 Encinal Station
Sunnyvale, California 94087
(408) 735-0871

	Retail	Your Price
P.F.S.II "NEW IMPROVED"	$125.00	$ 96.00
P.F.S. REPORT	95.00	74.00
P.F.S. GRAPH "NEW"	125.00	96.00
(Interfaces with P.F.S. and VisiCalc)		
VISICALC 3.3	250.00	225.00
VISITERM	100.00	80.00
VISIDEX	250.00	195.00
VISIPLOT	200.00	158.00
SUPER DISK COPY II	30.00	25.00
PRO EASYWRITER/MAILER COMBO	300.00	225.00
EASYSPELLER	175.00	137.00
THE DICTIONARY	99.50	74.00
CREATIVE FINANCING	195.00	143.00
REAL ESTATE ANALYZER II	195.00	143.00
THE BOOKKEEPER MASTER	89.95	66.00
THE HOME ACCOUNTANT	74.95	60.00
TYPING TUTOR II	24.95	19.95
SUPER DISK COPY III	30.00	26.00
APPLE 21	24.95	21.95
CRAPS	24.95	21.95
CROSSFIRE	29.95	24.97
SABOTAGE	24.95	21.95
THREE MILE ISLAND	39.95	33.97
STAR THIEF	29.95	23.97
Hardware		
EPSON MX 80 PRINTER	599.00	525.00
EPSON MX 100 PRINTER	995.00	799.00
OKIDATA 82A PRINTER		510.00
OKIDATA 83A PRINTER		799.00
PROMETHEUS VERSAcard	199.00	167.00
16K RAM CARD	169.00	105.00
AMDEK VIDEO 100 MONITOR	179.00	117.00
AMDEK VIDEO 300 MONITOR	249.00	208.00
AMDEK COLOR I MONITOR	449.00	384.00

Send check or money order. CA residents add 6% sales tax. Add $2.00 for postage. All items subject to availability. Send $.50 for our catalog or a free catalog with your order.

Olympic Sales Company

SERVING YOU SINCE 1947

Main Showroom & Offices:
216 South Oxford Avenue
Los Angeles, CA 90004

WE HONOR
VISA and MASTERCHARGE

TELEX: 67 34 77
OROER DESKS open 7 Oays a Week!
7:00 AM to 7:00 PM Mon thru Sat
Sunday Noon to 5:00 PM
Order Oesks: (213) 739-1130

TOLL-FREE (outside Calif.)	TOLL-FREE (within Calif.)
800-421-8045	800-252-2153

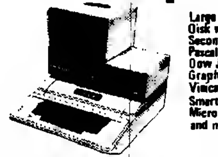
NO ONE UNOERSELLS OLYMPIC SALES

Write & request our new 112 pg catalog-please include $1.00 to defray postage & handling-includes many more items-from TV's to Watches!

All goods subject to availability, this ad supersedes all previous ads, we are not responsible for typographical errors; we will meet or beat any advertised prices if the competition has the goods on hand.
Minimum shipping and handling $4.95.
All orders subject to verification and acceptance.

apple computer

We are an authorized servicing Apple dealer for Apple II & III Immediate delivery on all models—we carry an enormous inventory of Apple products at all times!



Immediate delivery
16K—32K—48K—64K—Plus or Integer in stock!
NEW, IMPROVED APPLE III 128K VERSION

Large Inventory of:
Oisk with controller OOS 3.3
Second Oisk Orives
Pascal—Fortran—Cobol languages
Oow Jones & Quote reporter
Graphics Tablets
Visicalc for Apple II & III
Smarterm 80 column card
Micromodem II by OC Hayes
and more. . .

ATTENTION: Immediate delivery
WE ALSO CARRY SOFTWARE!
Personal Software
Peach Tree Software
Microsoft
Micropro
Innovative
American
System Plus
and more. . .

NOW IN STOCK!

HEWLETT PACKARD

2 NEW DELUXE CALCS FROM HP!
Slim, shirt-pocket styling

NEW! HP-11C
Advanced Programmable Scientific LCO *Retail $130.00* — *Your Cost:* **$114.95**

NEW! HP-12C
Advanced Programmable Financial LCO *Retail $150 00* — *Your Cost:* **$129.95**

	Retail	Your Cost
HP-125 New Microcomputer 84K CPU/Terminal/Keyboard	3750 00	3195.00
HP-85 Microcomputer—built-in printer/monitor	3250.00	2499.00
HP-83 Microcomputer—built-in monitor	2250.00	1799.00
HP-2831B Printer, dot mtrx (ask for optns)	3950.00	3199.00
HP82905A 80 col printer, dot matrix	945.00	755.00
HP-2601A Letter quality prter, daisy wheel		3495.00
HP-82901M Dual master (258KB disk drive)	2500.00	1999.00
Call us for the lowest prices on 7 disk drives	Call us	Call us
HP-41CV New! 2.2K bytes of memory	325 00	258.00
Card Reader for 41CV	215.00	171.00
Printer for 41CV	385.00	294.00
Ouad Ram	95.00	84.95
Optical Wand for 41CV	125.00	99.95
HP-41C Calculator	250 00	188.95
Memory mod. for 41C		26.95
HP-97 Progrmble printer	750 00	579.95
HP-34C Progrmble scientific	150 00	117.95
HP-38C Progrmble business R/E	150 00	119.95

We have the complete line of accessories, etc.

XEROX

Model 820-1 (5¼") — 8" $3750.00
64K COMPUTER & WORO PROCESSOR
AS LOW AS $2995.00
Special discount available to Schools & Institutions—Inquire! Required software add'l.

NEW—FAMOUS CORVUS OISK ORIVES—5, 10, 20 MEGABYTES with fantastic new OMNINET Network System
Call us for the best prices in the USA!

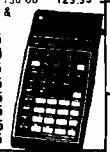
Texas Instruments Home Computer

New—1982 Model with full typewriter-style keyboard, TI-99/4A U/L case & more!
New KEYBOARO **$369.95**

New keyboard

	Retail	Your Cost
TI-99/4A New keyboard	$950 00	$369.95
10" color monitor for 99/4	650 00	319 95
32K Exp. mem module	399 95	314 95
Extended Basic, a MUST for 32K module	100 00	75 00
Speech synthesizer	150 00	129.95

We carry a large inventory of software, & accessories



	Your Cost
TI-30-2 LCS Stu Slide NEW	18.95
TI-35SP LCO SCIENTIFIC	22.50
TI-40 LCD Sci/NEW	28.95
TI-57 Prog. Scientific	39.95
TI-58C 480 Step Prog.	89.95
TI-59 960 Step. Prog.	179.95
PC-100C Print/Plot	149.95
LCD-Programmer/NEW	59.95

ATARI Computer

Large inventory of periphls, access. etc.

		Retail	Your Cost
400	SPECIAL PRICE! 16K	595 00	339.95
	No language inc., opt'l basic,		54.95
800	16K Computer	1080 00	759.95
	SPECIAL! ATARI 800 48K Computer	1250.00	869.95

OHIO SCIENTIFIC

CBPDF—48K Retail:$3495.00 Y/C: $3195.00
- Dual 8" Drives ● 64 col x 32 line/color
- .7 MIPS —*FAST!* ● Many more stnd features

Fortran & Pascal available
Many other OSI products—at discounted prices

NEW! From TI—Series 10 Personal Information Terminal *Retail* 995 00 *Your Cost* 795.00

PRINTERS

● DIABLO (Letter Quality)	Retail	Your Cost
630 R102 bi-directnal/tractors	2965 00	2699.00
1640K109 keyboard, tractors	3072 00	2899.95
630 RO Receive only	2710.00	2499.95
1650K136 keyboard/tractors	3220.00	2999.95
● CENTRONICS dot matrix		
700-9 Parallel, heavy duty	1460.00	1199.95
704-9 Serial, heavy duty	1795.00	1599.95
737-1 Parallel	995.00	799.95
737-3 Serial	1045.00	899.95
704-11 Parallel	1870.00	1695.00
P-1 Electrostatic	495.00	189.95
● PAPER TIGER		
46G	995.00	895.00
460G graphics	1094.00	969.95
560	1295.00	1099.00
560G graphics	1394 00	1195.00
445	795.00	695.00
445G	894 00	789.00
EPSON PRINTERS	Retail:	Your Cost
MX80	645.00	539.95
Optional Graftrax Chip 80		95.00
MX80 FT	745.00	659.95
MX80 + GRAFTRAX 80	695.00	579.95
MX80 FT + GRAFTRAX 80	795.00	689.95
MX100	995 00	789.95

WE ALSO HAVE . . .

● NOVATION Modems	Retail	Your Cost
CAT	199 95	159.95
D-CAT	199 95	159.95
APPLE-CAT Direct connect	349 95	314.95

SANYO MONITORS High resolution

13" Color (new) high quality	550 00	419.95
12" Green phosphorous	360 00	259.95
12" Black & white	340 00	239.95
15" Black & white	370 00	259.95
9" Black & white (the best seller)	235 00	169.95

AMDEK(Leedex)High Quality Monitors

100	12" B/W, 12 MHz	179.00	139.95
100-G	12" Green, 12 MHz	199.00	174.95
300-G	12" Green, 18 MHz	249.00	199.95
Color I	13" Color, NTSC comp. input, audio amp & speaker	449.00	339.95
Color II	13" Color, RBG input, hi res graphics, speaker	999 00	699.95

- HAZELTINE Video Display Terminals
- SHUGART Disk Drives
- OEC VT100 & VT103

Call us for your DISCOUNTED price TODAY!

Data Entry Options

a. Entry or change of specific data within the displayed year: Enter the location of the new data (row # shown at left of screen, item # shown at top of screen), followed by the new value of data. For example, to enter a value of 10 for TAM (item #2) of product category #3, enter:

```
3,2,10 (enter)
```

b. Entry or change of all values of a specific item # within the displayed year: Enter an "*" for the row, followed by the item # and a comma. The program will then prompt the user for the desired data value for each row (product category) by displaying the name of the product category, and the present value (PV). Three possible entries of data may be made: 1. the new value, 2. *enter*, which is accepted as a value of zero, and 3. "/", which retains the present value.

Example of entering data for TAM for all product categories:

```
*,2, (enter)
```

c. Entry or change of all values of data for a specific product category (row): Enter the row #, followed by an "*" for the item #, followed by a comma. The program will then prompt the user to enter data for each of the nine items for the specific product category designated. A value must be entered for each item request. The "enter" key with no preceding value is interpreted as a value of zero.

Example of requesting entry for all items for product category #14:

```
14,*, (enter)
```

d. Change of primary or secondary year: The value of the years for primary and secondary display pages is initialized at 1981 and 1986, respectively. Either of these years may be changed to any other year value. The display should be set to the year to be changed (pages 1 through 3 for primary year, pages 4 through 6 for secondary year). The enter command may then be used, as shown in the following example, which changes the year to 1989:

```
YR,,1989 (enter)
```

When any of the entry requests are completed, the program will always return to DISPLAY.

CALCULATE

The calculate routine which generates data within the displayed year, projects data from the primary year to the secondary year using an entered annual compound growth rate, or redistributes either the TAM or NSB of the displayed year to equal an entered top-down summation. The annual compound growth rate between items in the primary and secondary years may also be calculated.

Operation in this routine is indicated by an inverted-screen "C" in the lower left of the display. Entry into CALCULATE may be performed from either MAIN CONTROL or DISPLAY, but exit is always to DISPLAY.

Upon entry to CALCULATE, the display will prompt the user with the following request:

DATA,RATE,PROJCTN,SPREAD?

(continued)

***Table 2:* Commands Available from MAIN CONTROL**

Command	Routine	Action
D	DISPLAY	Program enters video display routine.
E	ENTER	Program enters data entry routine.
C	CALCULATE	Program enters calculate routine, where new data is determined based on previously entered data.
S		Calculates sums of all columns within the year that is presently displayed on video.
CLR	CLEAR	Permits clearing data in either primary or secondary year, as specified.
R		Displays previously calculated compound growth rates on video screen (see CALCULATE).
P		Data within vertical page last displayed is output to the printer, as described in PRINTER OUTPUT.
RP		Outputs previously calculated compound growth rates to the printer (see CALCULATE).
LD	LOAD	Loads yearly data previously stored on cassette tape.
SAVE	SAVE	Saves primary or secondary year data on cassette tape.
HEADER		Cassette tape is prepared to receive data for a year not previously stored. See "Tape Initialization" under SAVE.

***Table 3:* Commands Available from DISPLAY**

Command	Routine	Action
R		Same as command described under MAIN CONTROL
E	ENTER	Same as above
C	CALCULATE	Same as above
P		Same as above
S		Same as above
+		Special Sum — calculates sums of columns between specific rows (product lines) designated by user.
Q		Return to MAIN CONTROL.

The user response must be followed by an "enter" key.

Calculate Options

D Data: Calculates data within the displayed year based on existing data. Entry into this subroutine results in the following request on the display:

ROW(*), ITEM #?

The location of the desired parameter is entered using the row # shown on the left of the screen and the item # shown at the top of the screen. If an entire column is to be calculated, an * may be used in place of the row #. For example, to calculate the value of industry units (item 4) based on TAM and ASP, enter the following:

*,4 (enter)

The calculations for TAM and NSB may be based on a variety of constants. Whenever a request for these calculations is made, the computer requests more information:

CONSTANT (3), (4)&(5), (1)?

The item # to be held constant is entered using the "enter" key, and the calculation is performed.

D134 Data: Automatically performs a series of calculations comparable to "D" for items 1, 3, and 4 for all product categories (rows).

D178 Data: Automatically performs a series of calculations comparable to "D" for items 1, 7, and 8 for all product categories (rows).

R Rate: Calculates the annual compound growth rate between primary and secondary year data. Entry into this subroutine results in the display of the prompt:

ROW(OR "*"),ITEM?

The growth rate for a particular product category (row) and item may be entered, or all product categories for the specific item may be calculated by using "*" for the row #. The algorithm for calculating the compound growth is relatively slow, so an indication of which row is being calculated is shown to indicate that the computer is operating. At the completion of the calculation, the value(s) are shown on the display as:

PRODUCT LINE   YEAR(1)
YEAR(2)   RATE(%)

Appendix A

```
1 CLEAR
9 ' MPP
10 ' L.I.SUCKLE
12 ' 12/80
15 ' REV:10/19/81
20 ' INITIALIZATION
25 '*****ENTER NUMBER OF PRODUCT CATEGORIES IN LINE 30
30 PC=6
35 PT=PC+1:PM=INT((PC+3)/11)+1
40 DIM T(PT,18),H1$(9),H2$(9)
45 DIM PL$(PT),SM(18),RT(PT)
50 DIM TM(4)
60 H1$="PRODUCT CAT."
80 DATA NSB/TAM,TAM,MIX-T,UNITS,ASP
100 DATA NSB,MIX-N,UNITS,ASP
120 DATA % (1),(2),(3),(4),(5)
130 DATA (6),(7),(8),(9)
160 FOR N=1 TO 9
180 READ H1$(N)
200 NEXT N
220 FOR N=1 TO 9
240 READ H2$(N)
260 NEXT N
270 CLS
275 PG=1:PP=1
280 Y1=1981
290 Y2=1986
300 YR=Y1
320 SL=32*13
330 '*****ENTER THE NAMES OF PRODUCT CATEGORIES IN LINES 340-400
340 DATA WIDGETS,THINGOTS,DIPSIES,DOODLES,WHATCHA,MACALLETS
410 FOR N=1 TO PC
420 READ PL$(N)
430 NEXT N
440 FX=10
450 GOSUB 9000
460 FD=FN
470 PL$(PT)="TOP-DOWN TOTALS"
480 JF=0
1000 REM MAIN CONTROL
1010 PRINT@SL
1020 PRINT@SL,"m";
1040 INPUT QO$
1055 Q$=LEFT$(QO$,1)
1060 IF Q$="D" THEN 2000
1070 IF Q$="E" THEN 7000
1080 IF QO$="SAVE" THEN 4000
1090 IF QO$="LD" THEN 5000
1095 IF QO$="CLR" THEN 11000
1100 IF Q$="C" THEN 6000
1110 IF Q$="P" THEN 8000
1120 IF QO$="HEADER" THEN 10000
1125 IF QO$="RP" THEN 2700
1130 IF Q$="R" THEN L1=1:L2=29:GOSUB 6340
1140 IF Q$="S" THEN GOSUB 2500
1999 GOTO 1020
2000 REM LIST ROUTINE
2020 CLS
2025 IF PG>3 THEN YR=Y2 ELSE YR=Y1
2030 ON PG GOTO 2038,2040,2050,  2052,2054,2056
2038 ST=1:GOTO 2060
2040 ST=2;GOTO 2060
2050 ST=6:GOTO 2060
2052 ST=10:GOTO 2060
2054 ST=11:GOTO 2060
2056 ST=15
2060 LS=1+11*(PP-1)
2070 IF PC<LS+10 THEN LE=PC ELSE LE=LS+10
2080 S=ST
2090 PRINT@0,RIGHT$(STR$(YR),4)
2100 PRINT @32,"(YR)"
2110 IF PG=1 OR PG=4 THEN 2300
2120 FOR L=4 TO 25 STEP 7
2125 IF PG>3 THEN S1=S-9 ELSE S1=S
2130 PRINT@L," "+H1$(S1)
2140 PRINT@L+32," "+H2$(S1)
2150 K=64
2155 IF LS>PC THEN 2220
2160 FOR P=LS TO LE
2170 IF L=4 THEN PRINT@K,P
2175 T=INT(T(P,S)*100+.5)/100
2180 PRINT@L+K,T;
2190 K=K+32
2200 NEXT P
2210 IF PP<>PM THEN 2270
2220 IF L<>4 THEN 2260
2230 PRINT@K+64,PT:PRINT@K+68,T(PT,S)
2240 PRINT@K+96,"DLTA":IF FS=1 THEN PRINT@K+100,
     INT(100*(T(PT,S)-SM(S))+.5)/100
2250 PRINT@K+32,"SUM"
2260 IF FS=1 THEN PRINT@L+K+32,INT(SM(S)*100+.5)/100
2270 S=S+1
2280 NEXT L
2290 GOTO 2900
```

(Continu

Appendix A *(continued)*

```
2300 PRINT@5,H1$
2310 PRINT@25,H1$(1)
2330 PRINT@57,H2$(1)
2340 K=64
2345 IF LS>PC THEN 2420
2350 FOR P=LS TO LE
2360 PRINT@K,P
2370 PRINT@K+4,PL$(P)
2380 PRINT@K+25,INT(T(P,ST)*100+.5)/100
2390 K=K+32
2400 NEXT P
2410 IF PP<>PM THEN 2900
2420 PRINT@K+36,"SUMMATION"
2430 IF FS=1 THEN PRINT@K+57,INT(SM(ST)*100+.5)/100
2440 PRINT@K+64,PT;" TOP-DOWN TOTAL"
2450 PRINT @K+100,"DELTA"
2460 GOTO 2900
2470 PRINT@SL,"sum FIRST ROW,LAST ROW";
2480 INPUT JB,JE
24B5 GOSUB 2505
248B PP=3
2490 JF=1
2495 GOTO 2000
2500 PRINT@SL,"CALCULATING SUMS"
2501 JS=1:JE=PC
2502 JF=0
2505 FOR N=1 TO 18
2510 SM(N)=0
2520 NEXT N
2530 FOR N=JS TO JE
2532 SM(3)=BM(3)+T(N,3)
2535 SM(7)=SM(7)+T(N,7)
2536 SM(12)=SM(12)+T(N,12)
2537 SM(16)=SM(16)+T(N,16)
2540 FOR J=2 TO 17 STEP 2
2550 SM(J)=BM(J)+T(N,J)
2560 IF J=B THEN J=9
2570 NEXT J
2580 NEXT N
2590 FOR J=0 TO 9 STEP 9
2600 IF SM(2+J)<>0 THEN SM(1+J)=100*SM(6+J)/SM(2+J)
2610 IF SM(4+J)<>0 THEN SM(5+J)=BM(2+J)/SM(4+J)
2620 IF SM(8+J)<>0 THEN SM(9+J)=BM(6+J)/SM(8+J)
2630 NEXT J
2635 FS=1
2640 RETURN
2700 REM RATE PRINT
2710 PRINT#-2,TAB(4)H1$(IR);
2720 PRINT#-2,TAB(11)Y1;
2730 PRINT#-2,TAB(18)Y2;
2740 PRINT#-2,TAB(25) "RATE"
2750 FOR N=1 TO PC
2760 PRINT#-2,CHR$(0);
2770 PRINT#-2,N;LEFT$(PL$(N),8);
2773 DR=N:DI=IR:GOBUB9500
2774 PRINT#-2,TAB(11)D;
2777 DI=IR+9:GOSUB9500
27B0 PRINT#-2,TAB(1B)D;
2790 PRINT#-2,TAB(25)INT(RT(N)*10+.5)/10
2B00 NEXT N
2B10 GOTO 1000
2900 PRINT@503,"PG"PG;
2901 PRINT@SL,"d";
2902 IF JF=1 AND PP=3 THEN PRINT@SL,"special sum ROWS"JS;" TO";JE
2903 PRINT@SL,"d";
2904 X$=INKEY$
2905 IF X$="" THEN 2902
2906 IF X$="R" THEN L1=1:L2=PC:GOTO6340
2909 IF X$="+" THEN 2470
2910 IF X$="Q" THEN 1000
2911 IF X$="E" THEN 7000
2912 IF X$="C" THEN 6000
2913 IF X$="S" THEN GOSUB 2500
2914 IF X$="P" THEN B000
2915 X=VAL(X$)
2920 IF X>0 AND X<7 THEN PG=X
2925 Y$=CHR$(ABC(X$)+16)
2930 Y=VAL(Y$)
2935 IF Y>0 AND Y<=PM THEN PP=Y
2940 IF X$=CHR$(9) THEN IF PG<6 THEN PG=PG+1
2945 IF X$=CHR$(8) THEN IF PG>1 THEN PG=PG-1
2950 IF X$=CHR$(94) THEN IF PP>1THEN PP=PP-1
2955 IF X$=CHR$(10) THEN IF PP<PM THEN PP=PP+1
2965 GOTO 2020
3000 REM MULTIPLE CALC.
3010 R$="*"
3020 I=1:GOSUB 6530
3030 I=3:GOSUB 6530
3040 I=4:GOSUB 6530
3050 GOTO 2000
3060 R$="*"
3070 I=1:GOSUB 6530
30B0 I=7:GOSUB 6530
```

(Continued)

When the entire column has been calculated, a display page is filled and waits for any key input before displaying the next page. After the last data is displayed, any key entry returns the operation to DISPLAY. The latest rate calculation may be reviewed from either DISPLAY or MAIN DISPLAY by entering "R". The rate data may be output to the printer by entering "RP" from MAIN CONTROL.

P Projection: Calculates a new value of data in the secondary year based on existing data in the primary year and an entered annual compound growth rate in %. Entry into this subroutine results in a displayed prompt:

ROW(* OR S), ITEM, RATE?

The specific row #, and annual rate are entered, followed by the "enter" key. The calculated data is stored in the appropriate location in the secondary year data table. Completion of the calculation returns the program to DISPLAY.

Sample ENTRY:

3,2,10 (enter)

Several options for "row #" are'

* Data for all product categories under the specified item # will be calculated. Using an entry, as shown by example:

*,2, (enter)

will result in a display prompt for annual rate inputs in percent for each product category.

S Data for all product categories under the specified item # will be calculated with identical growth rates equal to that value entered. An example for items 2 and 10% is:

S,2,10 (enter)

The growth rate previously calculated with the "R" command under CALCULATE may be used in place of an entered value(s) for rate. This is performed by using "R" for the value of the rate. An example of projecting an entire column of data is:

*,2,R (enter)

S Spread: Redistributes the values of either TAM or NSB within the year last displayed so the total is equal to an entered top-down summation. Redistribution is based on product mix (item 3 or 7). Entry into this subroutine yields the prompt:

ITEM # (2 OR 6)?

Either 2 or 6 will be accepted, using the "enter" key, for spreading TAM or NSB, respectively. The program checks for a valid entry of top-down summation, and if one exists, the calculation is performed and the program returns to DISPLAY. If an invalid summation is found (zero), the program requests that a top-down entry be made and returns to MAIN CONTROL (indicated by inverted-screen "M").

CLEAR

Data for either the primary or secondary year may be reinitialized to zero by entering CLR from MAIN CONTROL. The program will prompt with:

CLEAR WHICH YEAR?

The year is entered (using "enter" key) and if it corresponds with either the primary or secondary year, all data in memory for that year will be cleared. If an invalid year is entered, the display will indicate:

YEAR NOT IN DISPLAY

Exit from the routine is to DISPLAY.

LOAD

The load routine retrieves data previously stored on cassette tape and loads it into memory. Access to this routine may only be performed from MAIN CONTROL. Entering LD results in the response:

LOAD WHICH YEAR INTO PAGES 1-3(1) OR 4-6(2)?

Pages 1-3 represent the primary year, and 4-6 the secondary year. For example, if data for 1984 is to be placed into the secondary year memory, the following entry would be made:

1984,2 (enter)

The user is then prompted to "REWIND TAPE & PLAY". After the data is loaded, the prompt, "TURN RECORDER OFF", is issued and the program returns to MAIN CONTROL.

Appendix A *(continued)*

```
3090 I=8:GOSUB 6530
3100 GOTO2000
3200 IF VAL(R$)<>0 THEN 3240
3205 R$="*"
3210 IF I=IR THEN 6911
3220 PRINT"CALCULATE RATES FOR ITEM";IR
3230 GOTO 1000
3240 Y3=VAL(R$)
3250 R$="*"
3270 GOSUB 6225
3280 Y2=Y3
3290 GOTO6911
4000 REM SAVE
4005 DL=200
4010 PRINT@SL,"SAVE WHICH YEAR";
4020 INPUT YS$
4022 YS=VAL(YS$)
4030 IF YS<>Y1 AND YS<>Y2 THEN  PRINT "DATA NOT AVAILABLE";:GOTO1000
4031 GOTO 4070
4035 IF RIGHT$(YS$,1)="X" THEN 4070
4040 PRINT @SL,"REWIND TAPE & 'PLAY'"
4050 FN$="H"+STR$(YS)
4060 SKIPF FN$
4070 PRINT@SL,"SET TO 'RECORD' & ENTER 'READY'"
4080 INPUT Q$
4090 IF Q$<>"READY" THEN 1000
4100 IF YS=Y1 THEN SF=1:EF=9
4110 IF YS=Y2 THEN SF=10:EF=18
4120 OPEN "O",-1,"F"+STR$(YS)
4125 RP=1
4160 FOR N=SF TO EF
4170 FOR K=1 TO PT
4175 ON RP GOTO 4180,4185
4180 PRINT#-1,T(K,N):GOTO4190
4185 INPUT#-1,T(K,N)
4190 NEXT K
4200 NEXT N
4210 CLOSE-1
4212 ON RP GOTO 4214,4220
4214 MOTORON
4216 FOR T=1 TO DL
4217 NEXT T
4218 MOTOROFF
4220 CLS
4230 PRINT@96,"TURN RECORDER OFF"
4240 GOTO 1000
5000 REM LOAD
5005 CLS
5010 PRINT@SL,"LOAD WHICH YEAR,"
5015 PRINT "INTO PAGES 1-3(1) OR 4-6(2)
5020 INPUT YS,YP
5030 IF YP<>1 AND YP<>2 THEN 1000
5035 IF YP=1 THEN SF=1:EF=9:Y1=YS
5037 IF YP=2 THEN SF=10:EF=18:Y2=YS
5040 CLS:PRINT@SL,"REWIND TAPE & 'PLAY'"
5050 OPEN"I",-1,"F"+STR$(YS)
5060 RP=2
5070 GOTO 4160
6000 REM CALCULATE
6010 PRINT@SL,"c DATA,RATE,PROJCTN,SPREAD";
6020 INPUT Q$
6022 FS=0
6025 IF Q$="D134" THEN 3010
6026 IF Q$="D178" THEN 3060
6030 Q$=LEFT$(Q$,1)
6040 IF Q$="D" THEN GOSUB 6500
6050 IF Q$="R" THEN GOSUB 6200:GOTO 1000
6060 IF Q$="P" THEN 6900
6070 IF Q$<>"S" THEN 2000
6100 REM SPREAD
6105 PRINT@SL
6110 PRINT@SL,"c ITEM # (2 OR 6)";
6120 INPUT I
6130 IF I<>2 AND I<>6 THEN 1000
6135 I1=I
6140 IF PG>3 THEN I=I+9
6142 IF T(PT,I)=0 THEN CLS:PRINT "ENTER TOP-DOWN ";H1$(I1):GOTO 1000
6145 GOSUB 2500
6150 D=T(PT,I)-SM(I)
6160 FOR N=1 TO PC
6165 IF SM(I)=0 THEN 6180
6170 T(N,I)=T(N,I)+D*T(N,I)/SM(I)
6180 NEXT N
6185 GOSUB 2500
6190 GOTO 2000
6200 REM RATE
6205 PRINT@SL
6210 PRINT@SL,"ROW(OR '*'),ITEM#";
6220 INPUT R$,I
6225 IR=I
6230 DY=Y2-Y1
6240 IF R$="*" THEN L1=1:L2=PC:GOTO 6270
6250 R=VAL(R$)
6260 L1=R:L2=R
```

(Continu

Appendix A *(continued)*

```
6270 FOR N=L1 TO L2
6275 IF T(N,I)=0 THEN 6330
6280 FX=T(N,I+9)/T(N,I)
6285 PRINT@SL,"CALCULATING FOR ROW ";N
6290 GOSUB 9000 'NAT.LN
6300 X=FN/DY
6310 GOSUB 9200'EXP
6320 RT(N)=(EX-1)*100
6330 NEXT N
6340 GOSUB 6360
6345 GOSUB 6420
6350 RETURN
6360 CLS
6370 PRINT@4,H1$(IR)
6380 PRINT@11,Y1
6390 PRINT@18,Y2
6400 PRINT@25,"RATE"
6410 L=32
6415 RETURN
6420 FOR R=L1 TO L2
6430 PRINT@L,R;PL$(R)
6440 PRINT@L+11,T(R,IR)
6450 PRINT@L+18,T(R,IR+9)
6460 PRINT@L+25,INT(RT(R)*10+.5)/10
6470 L=L+32
6472 IF L<480 THEN 6480
6474 Q$=INKEY$
6476 IF Q$="" THEN 6474
6478 GOSUB 6360
6480 NEXT R
6490 RETURN
6500 REM DATA
6505 PRINT@SL
6510 PRINT@SL,"c-data ROW(*),ITEM #";
6520 INPUT R$,I
6530 IF R$="*" THEN L1=1:L2=PC: GOTO6560
6540 R=VAL(R$)
6550 L1=R:L2=R
6560 I1=I
6570 ON I GOTO 6580,6590,6600,6610,6620,6630,6640,6650,6660
6580 V1=2:V2=6:EQ=4:GOTO 6670
6590 V1$="(3),":V2$="(4)&(5),":V3$="(1)":GOTO6750
6600 V1=2:EQ=3:GOSUB 2500:GOTO 6670
6610 V1=2:V2=5:EQ=1:GOTO6670
6620 V1=2:V2=4:EQ=1:GOTO6670
6630 V1$="(1),":V2$="(8)&(9),": V3$="(7)":GOTO 6750
6640 V1=6:EQ=3:GOSUB2500:GOTO6670
6650 V1=6:V2=9:EQ=1:GOTO6670
6660 V1=6:V2=8:EQ=1:GOTO6670
6670 IF PG>3 THEN V1=V1+9:V2=V2+9:I1=I1+9
6680 FOR R=L1 TO L2
6690 ON EQ GOTO 6695,6710,6715,6725,6728,6705
6695 IF T(R,V2)=0 THEN 6730
6700 T(R,I1)=T(R,V1)/T(R,V2):GOTO 6730
6705 T(R,I1)=T(R,V1)*T(R,V2)
6706 GOTO 6730
6710 T(R,I1)=T(R,V1)*T(30,I1)/100:GOTO6730
6715 IF SM(V1)=0 THEN 6730
6720 T(R,I1)=100*T(R,V1)SM(V1):FS=0
6721 GOTO 6730
6725 IF T(R,V1)=0 THEN 6730
6726 T(R,I1)=100*T(R,V2)/T(R,V1):GOTO6730
6728 T(R,I1)=T(R,V1)*T(R,V2)/100:GOTO 6730
6730 NEXT R
6740 RETURN
6750 PRINT@SL,"CONSTANT ";V1$;V2$;V3$
6760 INPUT V1
6770 IF V1=3 OR V1=7 THEN EQ=2: GOTO 6670
6773 IF V1=1 AND I1=6 THEN V2=2:EQ=5:GOTO 6670
6775 IF V1=1 THEN V2=6:EQ=4:GOTO 6670
6780 IF V1<>4 AND V1<>8 THEN 6750
6790 V2=V1+1
6800 EQ=6
6820 GOTO 6670
6900 REM PROJECTION
6901 W=0
6902 Y4=Y2
6905 PRINT@SL,"c projection ROW(* OR S), ITEM,  RATE";
6908 INPUT R$,I,RT$
6909 IF RT$="R" THEN 3200
6910 RT=VAL(RT$)
6911 IF R$<>"*" THEN 6917
6912 CLS:PRINT"ENTER RATE·FOR ";H1$(I)
6913 FOR W=1 TO PC
6914 IF RT$="R" THEN RT=RT(W):GOTO6916
6915 R$=STR$(W):PRINT W;PL$(W);:INPUT RT
6916 R$=STR$(W)
6917 R1=1+RT/100
6920 DY=ABS(Y2-Y1)
6925 S=1/R1
6930 FOR N=0 TO DY
6935 S=S*R1
6940 NEXT N
6945 IF Y2-Y1<0 THEN S=1/S
```

(Continued)

SAVE

The save routine stores data for a complete year on cassette tape. The routine must be entered from MAIN CONTROL. Entering SAVE will result in the response:

SAVE WHICH YEAR?

The year to be stored is entered (followed by the enter key). If the entered year does not correspond to either the primary or secondary years, the display will print "DATA NOT AVAILABLE" and return to MAIN CONTROL.

If a valid year is entered, the instruction "REWIND TAPE & PLAY" will be issued. The computer will search for the appropriate space on the tape assigned to that year (see Tape Initialization) and then prompt the user with "SET TO RECORD & ENTER 'READY' ". Following this instruction, the data is stored on the tape. The user is then instructed to "TURN RECORDER OFF" and control is returned to MAIN CONTROL.

Tape Initialization

Data is stored on a year-by-year basis, in a file named "F", followed by the year (e.g., "F1983"). The data for each year is spaced on the tape to allow sufficient room for updating by using a "header" file (named "H" followed by the year) preceding each data file. If a new data tape is being used, or if a different year is being stored than has been previously stored, the tape must be prepared to accept the data. This may be done with the following procedure:

1. Advance the data tape to that location where the new yearly data is to be stored.

2. Set the tape recorder to the record mode.

3. Return the program to the MAIN CONTROL routine.

4. Enter "HEADER".

5. Enter the year when requested.

The program will write the Header file and return to MAIN CONTROL. Continue with:

6. Enter "SAVE".

7. When requested with "SAVE WHICH YEAR", enter the year, followed immediately with an "X" (e.g., "1983X").

8. Enter "READY" when requested to do so.

9. Turn off the recorder when instructed.

The data has now been stored on tape and the program will return to MAIN CONTROL.

Summary

The Market Projection Program operates on a 16K Radio Shack Color Computer with Standard BASIC ROM, rather than the Extended BASIC ROM. Several subroutines were required to perform the mathematical operations of natural log and exponentiation. The program could easily be modified to operate with Extended BASIC by replacing the rather lengthy subroutines with their simple instruction

Appendix A *(continued)*

```
6950 IF R$="S" THEN L1=1:L2=PC: GOTO 6965
6955 R=VAL(R$)
6960 L1=R:L2=R
6965 FOR N= L1 TO L2
6970 T(N,I+9)=S*T(N,I)
6975 NEXT N
6980 IF W<>0 THEN NEXT W
6982 IF Y2<>Y4 THEN CLS:PRINT@495,Y4:GOTO2025
6985 GOTO 2000
7000 REM ENTER
7005 FS=0
7010 PRINT@SL,"# ROW(* OR YR),ITEM #(*),VALUE"
7015 PRINT"    (OR 'YR',,YEAR)";
7020 INPUT R$,I$,V
7021 IF R$="Q" THEN 1000
7023 IF I$="*" THEN 7300 ELSE I=VAL(I$)
7025 I1=I
7026 IF PG>3 THEN I=I+9
7030 IF R$="*" THEN 7090
7040 IF R$<>"YR" THEN 7070
7050 IF PG<4 THEN Y1=V ELSE Y2=V
7060 GOTO 7200
7070 R=VAL(R$)
7080 T(R,I)=V
7085 GOTO 7200
7090 CLS
7100 PRINT"ENTER VALUES FOR ";H1$(I1)
7110 FOR L=1 TO PC
7120 PRINT L;PL$(L);
7125 PRINT "  PV="T(L,I);:INPUT T$
7130 IF T$="/" THEN 7140
7135 T(L,I)=VAL(T$)
7140 NEXT L
7200 GOTO 2000
7300 R=VAL(R$)
7310 IF PG>3 THEN I1=10:I2=18 ELSE I1=1:I2=9
7320 CLS
7330 PRINT "E- ";PL$(R);YR
7340 FOR I=I1 TO I2
7350 PRINT H1$(I);
7360 INPUT T(R,I)
7370 NEXT I
7380 GOTO 2000
8000 REM PRINT
8010 IF YR=Y1 THEN F=0 ELSE F=9
8020 PRINT#-2,YR;H1$;
8030 PRINT#-2,TAB(22) H1$(1)
8050 FOR N=1 TO PC
8060 D1=INT(T(N,1+F)*100+.5)/100
8065 PRINT#-2,CHR$(0);
8070 PRINT#-2,N;PL$(N);
8080 PRINT#-2,TAB(24) D1
8090 NEXT N
8091 PRINT#-2
8092 PRINT#-2,"SUMMATION";
8093 PRINT#-2,TAB(24) INT(SM(1+F)*100+.5)/100
8094 PRINT#-2,PT;"TOP-DOWN TOTAL"
8095 PRINT#-2,"     DELTA"
8100 FOR D2=1 TO 5 STEP 4
8110 PRINT#-2
8120 PRINT#-2
8130 PRINT#-2,YR;H1$(D2+1);
8140 PRINT#-2,TAB(11) H1$(D2+2);
8150 PRINT#-2,TAB(18) H1$(D2+3);
8160 PRINT#-2,TAB(25)H1$(D2+4)
8170 FOR N=1 TO PC
8180 FOR K=1 TO 4
8190 I=K+F+D2
8200 TM(K)=INT(T(N,I)*100+.5)/100
8210 NEXT K
8215 PRINT#-2,CHR$(0);
8220 PRINT#-2,N;TM(1);
8230 PRINT#-2,TAB(11) TM(2);
8240 PRINT#-2,TAB(18) TM(3);
8250 PRINT#-2,TAB(24) TM(4)
8260 NEXT N
8270 PRINT#-2
8275 PRINT#-2,"SUM";
8280 FOR K=1 TO 4
8290 PRINT#-2,TAB((K-1)*7+3) INT(SM(K+F+D2)*100+.5)/100;
8300 NEXT K
8310 PRINT#-2
8320 PRINT#-2,PT;T(PT,F+D2+1)
8325 D3=INT((T(PT,F+D2+1)-SM(F+D2+1))*100+.5)/100
8330 PRINT#-2,"DLT";D3
8500 NEXT D2
8600 GOTO 2000
9000 REM LOG
9010 FM=1
9020 IF FX=<10 THEN 9060
9030 FX=FX/10
9040 FM=FM*10
9050 GOTO 9020
```

(Continu

Appendix A *(continued)*

```
 9060 GOSUB 9110
 9070 IF FM<10 THEN RETURN
 9080 FN=FN+FD
 9090 FM=FM/10
 9100 GOTO 9070
 9110 FK=(FX-1)/(FX+1)
 9120 F2=FK*FK
 9130 FZ=1/51
 9140 FOR F=49 TO 1 STEP-2
 9150 FZ=1/F+FZ*F2
 9160 NEXT F
 9170 FN=FZ*FK*2
 9180 RETURN
 9200 REM EXP
 9210 Z=1
 9220 EX=0
 9230 FOR XE=1 TO 50
 9240 EX=EX+Z*X/XE
 9245 Z=Z*X/XE
 9250 NEXT XE
 9255 EX=EX+1
 9260 RETURN
 9500 REM ROUND-OFF CALCULATE
 9510 D=INT(T(DR,DI)*100+.5)/100
 9520 RETURN
10000 REM HEADER PREPARE
10010 CLS
10020 PRINT@SL,"YEAR";
10030 INPUT YS
10040 OPEN "O",-1,"H"+STR$(YS)
10050 PRINT#-1,"HEADER",YS
10060 CLOSE-1
10080 DL=500
10090 Y1=YS
10100 GOTO 1000
11000 PRINT@SL,"CLEAR WHICH YEAR"
11010 INPUT Y
11020 IF Y=Y1 THEN K1=1:K2=9:GOTO 11100
11030 IF Y=Y2 THEN K1=10:K2=18:GOTO 11100
11040 PRINT "YEAR NOT IN DISPLAY":GOTO 2000
11100 FOR N=1 TO PT
11110 FOR I=K1 TO K2
11120 T(N,I)=0
11130 NEXT I
11140 NEXT N
11150 GOTO 2000
```

counterpart. There are also other command substitutions, such as providing formatted output, which would shorten the program. The difficulty with performing these changes, of course, is that the program will no longer run with a computer which only has the Standard BASIC ROM.

The listing for the program is given as Appendix A. To customize the program to specific market areas, the product categories of interest should be entered on lines 340 through 400. The number of categories should be entered on line 30 as the value of PC (product categories). The program will accept any number of categories; however, the amount of memory required for operation will increase or decrease according to the category quantity.

The author may be contacted at Motorola Inc., Semi Conductor Products Sector, 5005 East McDowell Rd., Phoenix, Arizona 85008.

MICRO

PET Vet

By Loren Wright

Carl Moser's MAE and ASM-TED

MAE and ASM-TED have been around since the early days of the PET. Other assemblers have come and gone or faded into obscurity, but few others have been able to match the prices, features, and support of these assemblers.

Both are written in machine language. ASM-TED is primarily intended to operate in a cassette environment with 16K or more. MAE requires a disk and 32K. The syntax and operation are nearly identical. Both are available for a number of other 6502 machines, which comes in handy when transferring programs from one machine to another.

MAE is a macro assembler. Macros allow you to assign a name to a frequently used sequence of instructions. Each time you want to include this sequence later in your program, you instruct the assembler to expand the macro you name. This can save a lot of time on large programming projects.

Another powerful feature of MAE is conditional assembly. One popular application of this capability is to allow one source listing to generate different object files, depending on which ROM set you have.

Another feature is interactive assembly, where the assembler pauses to get some input from the user before it continues assembling. MAE supports labels up to ten characters long, and the editor allows extensive, convenient file editing.

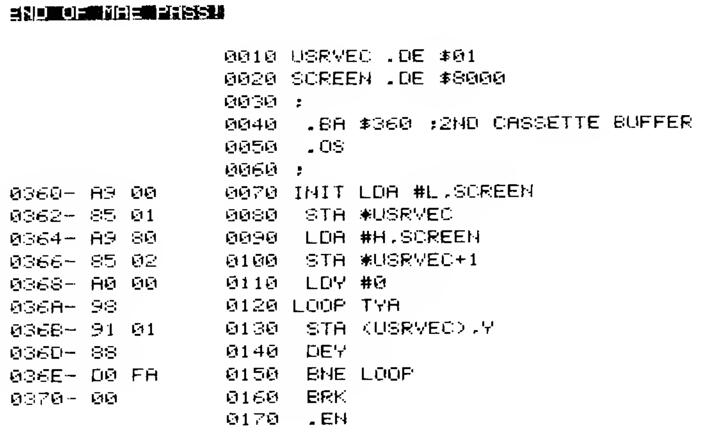
MAE coexists with BASIC, an extended monitor, and your assembly program, if you haven't decided to assemble it on top of the assembler! To handle such problems and to allow greater programming flexibility, a relocating loader program is included. The loader acts upon a relocatable object file, which is easily created by the assembler. An enhanced DOS wedge program is included, which allows you to enter the assembler, loader, or extended monitor (also included on the disk), with a two-letter abbreviation, such as 'AW' for 'assembler warm-start.'

The assembler offers a wide variety of pseudo-ops, including codes to handle data, address, and word storage; listing generation; assembly from a sequence of modules with a control file (not implemented in ASM/TED); as well as the macro, conditional assembly, and interactive assembly commands, and the more usual pseudo-ops.

Unlike the standard three-letter syntax, MAE uses two-letter codes, preceded with a period. For instance, instead of ORG, MAE uses .BA (begin assembly.)

There are other deviations from normal 6502 assembler syntax. Page zero labels must be preceded with an asterisk. To specify the high byte of a label, the syntax is: #H, Label. It is also important to distinguish in the equate portion of the file between label references internal and external to the progam, particularly if the relocating loader is to be used. These variations, learned quickly, were included to increase the speed and flexibility of MAE and the relocating loader.

Sample MAE Listing



```
END OF MAE PASS!

                  0010 USRVEC .DE $01
                  0020 SCREEN .DE $8000
                  0030 ;
                  0040  .BA $360 ;2ND CASSETTE BUFFER
                  0050  .OS
                  0060 ;
0360- A9 00       0070 INIT LDA #L.SCREEN
0362- 85 01       0080  STA *USRVEC
0364- A9 80       0090  LDA #H.SCREEN
0366- 85 02       0100  STA *USRVEC+1
0368- A0 00       0110  LDY #0
036A- 98          0120 LOOP TYA
036B- 91 01       0130  STA (USRVEC),Y
036D- 88          0140  DEY
036E- D0 FA       0150  BNE LOOP
0370- 00          0160  BRK
                  0170  .EN
```

Several utility and sample programs are included on the disk. The manual is thorough, including examples of macro, conditional, and interactive assembly. EHS publishes a newsletter on an irregular basis to keep users up on improvements, applications, and other information of interest to MAE owners. Finally, there is a large and very active users' group dedicated to Carl Moser's assemblers and assembly-language programming on the PET. They maintain a large disk library of assembly and other PET programs.

ATUG Disk Exchange
c/o Brent Anderson
200 S. Century
Rantoul, IL 61866
(217) 893-4577

MAE and ASM-TED are available from Eastern House Software (3239 Linda Drive, Winston-Salem, NC 27106). MAE comes on disk for $169.95, and ASM-TED on cassette for $49.95. Specify BASIC ROM and disk format.

The Transactor is Back!

After a brief absence, *The Transactor* is back. Commodore decided to discontinue publishing it after Volume 3. Karl Hildon has left Commodore to continue as editor of the new Transactor, now under the auspices of Canadian Micro Distributors. The new publication will undergo some changes, including a glossier presentation, advertising, and a more reference-oriented format. Most of the same authors will continue to contribute. A subscription to Volume 4 is $15 US or Canadian.

The Transactor
Canadian Micro Distributors
500 Steeles Avenue
Milton, Ontario L9T 3P7
Canada

Midnight Software Gazette and The Paper Merge

The *Midnight Software Gazette*, now published by the Central Illinois

PET Users' Group on a SASE basis, and *The Paper*, now published by Ralph Bressler, have merged. The name of the new publication will be determined by the results of a reader contest, in which the grand prize will be a VIC. Current *Paper* subscriptions will be fulfilled with issues of the new publication. A six-issue subscription is $20 — US, Canada, and Mexico; $30 — foreign surface; and $40 — foreign air mail. As with the *Midnight Software Gazette*, the new publication will be mailed First Class. Advertising is $25 per quarter page.

Midnight Software Gazette
(for the moment)
c/o Jim Oldfield
635 Maple Street
Mt. Zion, IL 62549

Other Commodore-oriented Publications

The Code Works has decided to discontinue publication of its famous *Cursor* cassette magazine until the new Commodore-64 is released. In the meantime, they will publish a newsletter. Another new Commodore-oriented newsletter is being published by Roger Olanson. Further information was not available at press time.

Strictly Commodore
47 Coachwood Place, N.W.
Calgary, Alberta T3H 1E1
Canada

The Whole PET Catalog Now Available

The *Midnight Software Gazette* has published a book called *The Whole PET Catalog*, which will include the contents of all of the first seven issues of *Midnight*, plus what would have been issue number 8. The material has been updated, edited, and reorganized. In addition there will be full listings of the contents of both the ATUG and TPUG libraries. The price for *The Whole PET Catalog* is $10 from the above address or it may be obtained from AB Computers (252 Bethlehem Pike, Colinar, PA 18915, (215) 822-7727), Professional Computer Sales (117 Skyway Avenue, Toronto, Ontario, Canada, (416) 675-7818), and other dealers. *Midnight* has concentrated on reviews and news, so it will be useful to see it all in one place.

Patch for 2031 with Upgrade BASIC

Willi Kusche (creator of KMMM Pascal, reviewed here in the January issue) of Bellwamr, NJ reports a problem in the operation of the CBM 2031 single disk drive with Upgrade ROM (3.0) PETs. He has provided the following patches to the $F000 ROM:

```
F17F    4C ED FF
F182    EA

FFED    AD 40 E8
FFF0    29 FB
FFF2    8d 40 E8
FFF5    A9 5F
FFF8    4C 87 F1
```

You can program a substitute EPROM yourself, or he will provide one for $15:

WILSERV INDUSTRIES
P.O. Box 456
Bellwamr, NJ 08031

Double-Entry Formatting for Your Checkbook on the C1P

by Leo Jankowski

A simple checkbook balancing program for the C1P is presented. The program is easily modified for other applications and computer environments.

Double Entry
requires:
OSI C1P
Cassette unit (optional)
Epson MX-80 (optional)

A drawback to many checkbook balancing programs is that they are more difficult to use than the pencil and paper approach! After all, you just need a credits column, a debits column, their totals, and difference.

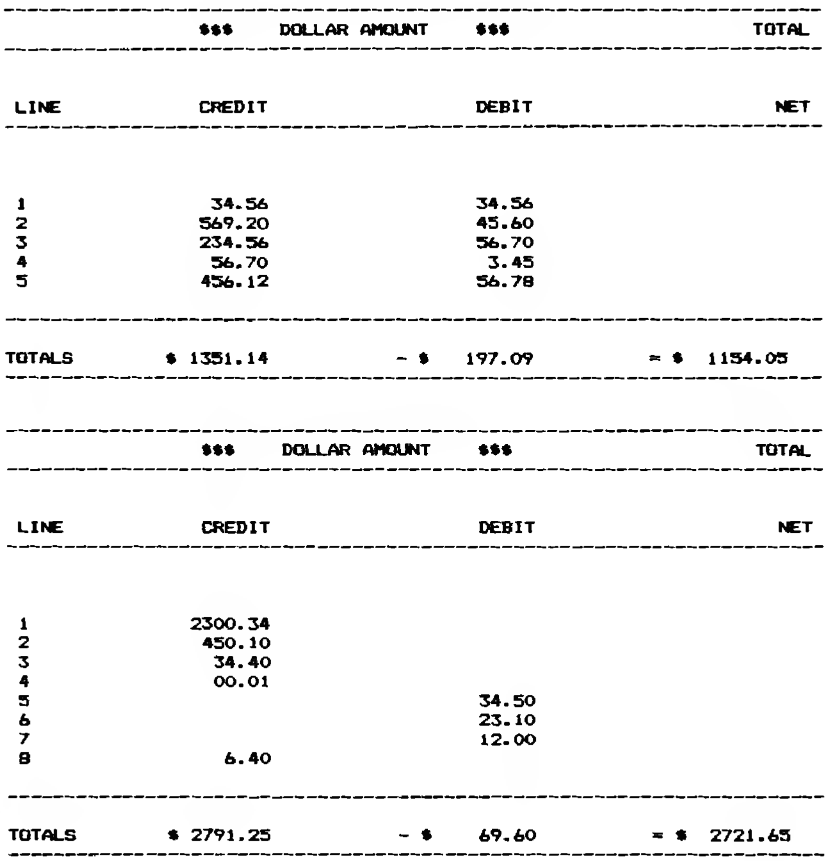
This program, designed primarily for use with a printer, provides this information. As is, the program is not concerned with check numbers, but if needed, they could be input as another column of (unformatted!) figures. The program could be easily adapted for other uses that require simple double-entry bookkeeping. By changing the headings to, say, GROSS PAY and TAX, a record could be made of paychecks and tax-bites due.

The user is given a choice of one entry per line or two. Every line used is numbered. Entries can be positive or negative, and in any format. Leading zeros can be omitted. The decimal point is not mandatory for dollar-only amounts. The only drawback is that the largest amount or total must be less than or equal to 9999.99. The C1P rounds off to seven significant figures or more, which introduces small arithmetical errors.

No entry-error checking routine is provided. Peculiar printouts will follow entries like 45.006! An error routine would have made the program unnecessarily complicated. The character codes in line 3080 are for the Epson MX-80. All lines containing REMs can be removed without affecting the program. Also, lines 3070 to 9998, and line 140, could be omitted without affecting the guts of the program.

Changing line 170 to 170 READ CA, allows you to use the program with DATA statements that have been added to the listing or that are to be read in from tape.

Sample Runs



```
-------------------------------------------------------------------
               $$$    DOLLAR AMOUNT    $$$                    TOTAL
-------------------------------------------------------------------


LINE           CREDIT                  DEBIT                    NET
-------------------------------------------------------------------



1               34.56                  34.56
2              569.20                  45.60
3              234.56                  56.70
4               56.70                   3.45
5              456.12                  56.78

-------------------------------------------------------------------

TOTALS       $ 1351.14         - $    197.09          = $   1154.05
-------------------------------------------------------------------

-------------------------------------------------------------------
               $$$    DOLLAR AMOUNT    $$$                    TOTAL
-------------------------------------------------------------------


LINE           CREDIT                  DEBIT                    NET
-------------------------------------------------------------------



1             2300.34
2              450.10
3               34.40
4               00.01
5                                      34.50
6                                      23.10
7                                      12.00
8                6.40

-------------------------------------------------------------------

TOTALS       $ 2791.25         - $     69.60          = $   2721.65
-------------------------------------------------------------------
```

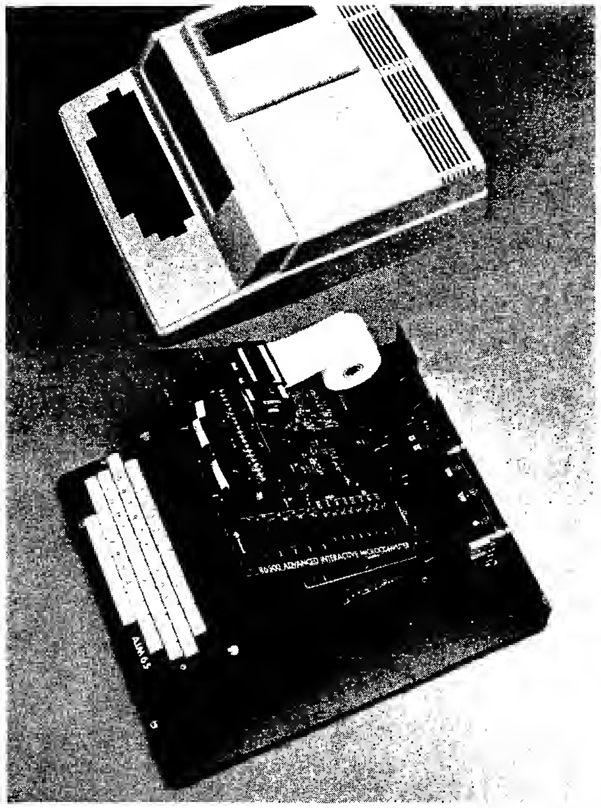
Mr. Jankowski has been involved with computers for 12 years. He now owns a Superboard and has several projects underway, including writing programs to teach Statistics and to keep stock records in farming. He may be contacted at Otaio RD1, Timaru, New Zealand.

Important Variables

C$	clear screen
CO	column
EN$	one entry per line or two
FF	counter for Form Feed
FL	form length
MU$	money string to be formatted
N	line number
NSUM	negative, 2nd column sum
PSUM	positive, first column sum
POKE 517,255	printer ON
POKE 517,0	printer OFF

Listing 1

```
10 POKE515,0:REM turn off LOAD.
90 EN$="Y"
100 FF=9:CO=22:FL=61:BL$=CHR$(32)
110 C$=CHR$(127):L$=CHR$(10):CR$=CHR$(13)
120 FH$="CREDIT":SH$="DEBIT":TH$="TOTAL":XH$="NET"
125 :
130 :
135 REM Go for Instructions & print headings.
140 GOSUB4000:GOSUB3070
141 :
145 :
146 REM Turn Printer OFF & ask for Input.
150 CO=22:POKE517,0
170 INPUT" Cash amount";CA:PRINT
180 CENTS=CASH*100
190 IFCA>0THENPSUM=PSUM+CENTS:GOSUB700:GOTO150
200 IFCA<0THENNSUM=NSUM+CENTS:CO=45:GOSUB700:GOTO150
210 :
220 REM CA must be 0
240 REM Turn Printer ON.
250 POKE517,255:PRINTL$
300 GOSUB2000:GOTO900
400 :
500 :
600 REM Decide where to print next amount.
700 POKE517,255
710 IF EN$="N"THENPRINTCR$:GOTO760
720 IFCA>0THENPRINTCR$:GOTO760
750 IFPE>0ANDCA<0THENCO=22:GOTO805
755 CO=22:IF PE<0ANDCA<0THENPRINTCR$:CO=45:GOTO760
758 CO=45
759 REM Line number.
760 N=N+1:PRINTN;
800 :
801 REM Format the cash amount.
805 POKE517,0
808 MUNNY$=STR$(CE):L=LEN(MUNNY$)
810 IFL=3THENL=5:MU$="00"+RIGHT$(MU$,2):GOTO855
820 IFL=2THENL=4:MU$="000"+RIGHT$(MU$,1):GOTO855
830 IFCE<0ANDL=3THENMU$="00"+RIGHT$(MU$,2):GOTO855
840 IFCE<0ANDL=2THENMU$="000"+RIGHT$(MU$,1):GOTO855
845 MU$=RIGHT$(MU$,L-1)
855 L=LEN(MU$)
870 POKE517,255
872 FF=FF+1:IF FF=FL THEN PRINTCHR$(12)
874 REM Form-feed ?
875 FF=FF+1:IF FF=FL THEN PRINTCHR$(12)
876 REM Print the formatted amount.
880 PRINTTAB(CO-L)LEFT$(MU$,L-2)"."RIGHT$(MU$,2);
890 PE=CA:RETURN
895 :
896 :
897 REM Last amount was 0, so now print totals.
900 PRINTBL$:PRINT"TOTALS";
905 CENTS=PSUM:PRINTTAB(14)"$";:GOSUB805
910 CO=45:CENTS=NSUM:PRINTTAB(34)"- $";:GOSUB805
920 CO=67:CENTS=PSUM+NSUM:PRINTTAB(56)"= $";
927 GOSUB805:PRINTBL$
928 GOSUB2000
935 PRINTCHR$(7):PRINTCHR$(12)
940 POKE517,0:GOTO9999
1000 :
1500 :
1600 REM Print a line.
2000 FORX=1TO71:PRINT"-";:NEXT:PRINTBL$:RETURN
2010 :
2020 :
3050 REM Print headings.
3070 POKE517,255
3075 REM Commands to EPSON 80 Printer.
3080 PRINTCR$:PRINTCHR$(27);:PRINTCHR$(69)
3090 GOSUB2000
3100 PRINTTAB(17)"$$$    DOLLAR AMOUNT    $$$";
3110 PRINTTAB(65)TH$
3120 GOSUB2000:PRINTL$
3130 PRINT" LINE"TAB(17)FH$TAB(41)SH$TAB(67)XH$
3140 GOSUB2000:PRINTL$
3150 POKE517,0:RETURN
3500 :
3600 :
3610 REM Instructions.
4000 PRINTC$
4010 PRINT" When prompted enter":PRINT
4020 PRINT" your money amounts.":PRINT
4030 PRINT" They can be negative"
4035 PRINT:PRINT" or positive."
4040 PRINT:PRINT:PRINT" Last amount must be 0."
4050 PRINT:PRINT:PRINT:PRINT
4060 INPUT" Two entries per line";EN$
4070 IF LEFT$(EN$,1)="N"THENEN$="N"
4080 PRINTC$:RETURN
4090 :
9998 DATA 0
9999 END
```



ARK COMPUTING

LOWERS THE BOOM ON HIGH PRICES

SPECIALS

16K Ram Board	$79.95
80 Column Board (Wiz-80)	$195.00
Parallel Printer Interface (W/Cable) (compatible with Pascal, Basic, CP/M)	$59.95
Z-80 Softcard	$259.95
Joyport	$59.95
ARK Special (includes Wiz-80, Lazer Keyboard Plus, Lazer Lower Case Plus)	$295.00
On-Line Systems: General Manager	$99.95
Diskettes w/hubring (10)	$19.95
Hi-Res Secrets	$84.95
Expediter II	$59.95
The Dictionary	$69.95
Microsoft 16K Ram Card	$99.95
Time Zone	$69.95
Lazer Keyboard Plus	$69.95
Lazer Lower Case Plus	$44.95
Lazer Lower Case Plus II	$19.95
Anix 1.0	$34.95
Lazer Pascal	$29.95
Anix-Pac (Anix, Pascal, Sources)	$99.95
Using 6502 Assembly Language	$14.95
Datamost Joystick	$39.95
Datamost Expandaport	$49.95

GAMES

Softporn Adventure (no grapnics)	$29.95/$19.95
#0 Mission Asleroids	$19.95/$14.95
#1 Myslery House	$24.95/$18.95
#2 Wizard & Princess	$32.95/$18.95
#3 Cranston Manor	$34.95/$24.95
#4 Ulyesses & the Golden Fleece	$34.95/$24.95
#5 Time Zone	$99.95/$69.95
Hi-res Cribbage	$24.95/$18.95
Hi-res Soccer	$29.95/$19.95
Hi-res football	$39.95/$29.95
Sabotage	$24.95/$19.95
Jawbreaker	$29.95/$19.95
Threshold	$39.95/$24.95
Missle Defense	$29.95/$19.95
Crossfire	$29.95/$19.95
Pegasus	$29.95/$19.95
Warp Destroyer	$29.95/$21.95
Star Crusier	$24.95/$19.95
Adventure	$29.95/$21.95
Bolh Barrels	$24.95/$19.95
Cyber Strike	$39.95/$29.95
Phanloms Five	$29.95/$21.95
Space Eggs	$29.95/$19.95
Autobann	$29.95/$21.95
Pulsar II	$29.95/$21.95
Orbitron	$29.95/$21.95
Gamma Goblins	$29.95/$19.95
Gorgon	$39.95/$29.95
Sneakers	$29.95/$19.95
EPOCH	$34.95/$24.95
Cops & Robbers	$34.95/$24.95
Oulpost	$29.95$19.95
Dark Forest	$29.95/$19.95
Beer Run	$29.95/$19.95
Borg	$29.95/$19.95
Joy Port w/foosball	$74.95/$59.95
Hadron	$34.95/$24.95
Twerps	$29.95/$19.95
Computer Foosball	$29.95/$19.95
Wizardry	$49.95/$39.95
Galactic Attack	$29.95/$21.95
Minator	$34.95/$24.95
Oylmpic Decathlon	$29.95/$21.95
Three Mile Island	$39.95/$29.95
ABM	$24.95/$19.95
Robot Wars	$39.95/$29.95
Global War	$24.95/$19.95
Castle Wollenstein	$29.95/$21.95
Falcons	$29.95/$21.95
Suicide	$29.95/$21.95
Grand Prix	$29.95/$21.95
The Best of Muse	$39.95/$29.95
Flight Simulator	$33.50/$26.95
Dungeon Campaign	$17.50/$14.95
Odyssey	$29.95/$21.95
Escape From Arcturus	$24.95/$19.95
Palace in Thunderland	$24.95/$19.95
MAD Venture	$24.95/$19.95
Roach Motel	$34.95/$26.95
English SAT #1	$29.95/$21.95
U S Constitution	$29.95/$21.95
Upper Reacnes of Apshai	$19.95/$15.95
The Keys to Acheron	$19.95/$15.95
Datestones of Ryn	$19.95/$15.95
Morloc's Tower	$19.95/$15.95
Ricochet	$19.95/$15.95
Starfleet Orion	$24.95/19.95
Invasion Onon	$24.95/$19.95
The Dragon's Eye	$24.95/$19.95
Tues. Morning Quarterback	$29.95/$21.95
Jabbertalky	$29.95/21.95
Sorcerer of Siva	$29.95/$21.95
Rescue at Rigel	$29.95/$21.95
Crush, Crumble, and Chomp	$29.95/$21.95
Temple of Apshai	$39.95/$29.95
Hellfire Warrior	$39.95/$29.95
Star Warrior	$39.95/$29.96
Hi-res golf	$29.95/$21.95
Race for Midnight	$29.95/$21.95
Midnight Music	$34.95/$24.95
Star Blazer	$31.95/$24.95
Apple Panic	$29.95/$19.95
Space Quark	$29.95/$19.96
Red Alert	$29.95/$19.96
Track Attack	$29.95/$19.95
Genetic Drift	$29.95/$19.95
Arcade Machine	$44.95/$29.95
Raster Blaster	$29.95/$21.95
Trilogy of Games	$29.95/$21.95
Space Album	$39.95/$29.95
Fender Bender	$24.95/$19.95
3-D Graphics	$39.95/$29.95
Akalabeth	$34.95/$24.95
Appleoids	$29.95/$21.95
Ultima	$39.95/$29.95
LA Land Monopoly	$29.95/$21.95
Hyperspace Wars	$29.95/$21.95
3-D Skiing	$24.95/$19.95
Torpedo Terror	$24.95/$19.95
Computer Bingo	$24.95/$19.95
Kaves Karkhan	$49.95/$34.95
Dragon Fire	$49.95/$34.95
Rings of Saturn	$39.95/$29.95
Alkemstone	$39.95/$29.95
Snack Attack	$29.95/$19.95
Casino	$39.95/$24.95
Thief	$29.95/$19.95
County Fair	$29.95/$19.95
Swashbuckler	$34.95/$24.95
Firebird	$29.95/$19.95
Russki Duck	$34.95/$25.95
Horizon V	$34.95/$25.95
Sargon II	$34.95/$25.95
Reversal	$34.95/$25.95
Zork	
Zork II	$39.95/$29.95
Pool 1.5	$34.95/$24.95
Shuffleboard	$29.95/$21.95
Trick Shot	$39.95/$29.95
Crossword Magic	$49.95/$39.95
Master Type (hi-res)	$39.95/$29.95
Dogfight	$29.95/$21.95
Crown of Arthain	$34.95/$26.95

UTILITIES

Hi-Res Secrets (D. Fudge)	**$124.95/$84.95**
Super Shape Drg & Animate	$34.95/$24.95
The Creative Tool Box	$44.95/$34.95
Applesoft Compiler	$175.00/129.95
Datadex	$150.00/$99.95
Using 6502 Assembly Lang	$19.95/$14.95
ANIX (UNIX-like Oper sys)	$49.95/$39.95
Lazer Pascal	$39.95/$29.95
DOSOURCE 3.3	$39.95/$24.95
Painter Power	$39.95/$29.95
The Voice	$39.95/$29.95
E Z Draw	$49.95/$39.95
Pascal Graphics Editor	$99.95/$69.95
ACE	$39.95/$29.95
Apple-DOC	$49.95/$39.95
List Master	$39.95/$29.95
ASCII Express	$79.95/$59.95
Z-Term (req's Z-80 Card)	$99.95/$89.95
Z-Term Pro	$149.95/$99.95
Speed Star	$134.95/$99.95
On-Line	$89.95/$69.95
DB Master	$229.00/$159.95
LISA v2.5	$79.95/$59.95
LISA Educational Pack	$119.95/$79.95
Speed/ASM	$39.95/$29.95
Expediter II	$99.95/$59.95
Disk Organizer II	$29.95/$21.95
Applesoft Plus	$24.95/$19.95
Applesoft Optimizer	$24.95/$19.95
Disk Recovery	$29.95/$21.95
Multi-disk Catalog	$24.95/$19.95
Back it up	$59.95/$39.95
Image Printer	$29.95/$29.95
Pascal Lower Case	$24.95/$19.95
Dos Plus	$24.95/$19.95
The Original Quick Loader	$24.95/$19.95
Typing Tutor	$24.95/$19.95
Fortran	$195.00/$139.95
A L D S	$125.00/$89.95
Basic Compiler	$395.00/$295.00
Cobol	$750.00/$495.00
TASC	$175.00/$129.95
Data Plot	$59.95/$44.95
DB Utility Pack	$99.00/$69.95

HARDWARE

12" hi-res green display	$285.00/$159.95
12" lo-res color display	$430.00/$349.95
12" hi-res RGB color display	$1095.00/$895.95
Apple Cat II	$389.00/$329.00
Apple Clock	$280.00/$195.95
Supertalker SD2000	$199.00/$149.95
Romplus	$155.00/$124.95
Romwriter	$175.00/$139.95
Music System	$395.00/$295.95
A/D+D/A	$350.00/$249.95
Expansion Chassis (125v)	$750.00/$495.00
CPS Multifunction Card	$239.00/$149.95
RAM Plus + (32 RAM board w/16K installed)	$189.00/$139.95
COPYROM	$55.00/$44.95
Keyboard Filter ROM	$55.00/$44.95
13 Key Keypad (new or old)	$125.00/$94.95
Bar code reader	$195.00/$145.95
Softkey (15-key user-defin)	$150.00/$119.95
Asynchronous Serial	$179.95/$129.95
Datamost Joystick	$59.95/$39.95
Datamost Expandaport	$69.95/$49.95
Micromodem	$379.00/$269.00
Smartmodem (RS-232)	$279.00/$199.00
LowerCase + (Rv 6 & earlier)	$64.95/$44.95
Lower Case + II (Rv 7 & Ltr)	$24.95/$14.95
Lower Case + III (Rv 7 & Ltr)	$49.95/$34.95
Keyboard #Plus	$99.95/$69.95
Character Set + Plus	$24.95/$19.95
Sup R' Term 80-col board	$395.00/$295.00
Sup R' Switcher (pwr suppl)	$295.00/$195.00
A35 35-track disk drive w/o controller	$479.00/$379.00
A35 35-track disk drive w/ controller	$579.00/$449.00
A40 40-track disk drive w/o controller	$449.00/$359.00
A40 40-track disk drive w/ controller	$549.00/$449.00
A70 70-track disk drive w/o controller	$599.00/$499.00
A70 70-track disk drive w/ controller	$699.00/$599.00

BUSINESS SOFTWARE

Mail Label & Filing System	$74.95/$49.95
Payroll	$395.00/$295.00
General Ledger (w/payables)	$495.00/$395.00
Home Money Minder	$34.95/$24.95
The Mail Room	$34.95/$24.95
CPA I General Ledger	$249.95/$169.95
CPA II Accounts Receivable	$249.95/$169.95
CPA III Accounts Payable	$249.95/$169.95
CPA IV Payroll	$249.95/$169.95
CPA V Property Manage	$249.95/$169.95
The Home Accountant	$74.95/$54.95
Word Star	$375.00/$195.00
Supersort	$200.00/124.95
Mail Merger	$125.00/$79.95
Data Star	$300.00/$195.00
Spellstar	$200.00/$124.95
Calc star	$200.00/$124.95
Visicalc 3.3	$250.00/$179.95
PFS (Personal Filing System)	$125.00/$89.95
PFS Report	$95.00/$69.95
The Correspondent	$59.95/$49.95
Mailing List Database	$49.95/$39.95
Higher Text II	$39.95/$29.95
Directory Manager v 2	$29.95/$21.95
Real Estate Analysis Pgm	**30% OFF**
PIE WRITER	**(call for details)**
Easy Mover 40-column	$49.95/$39.95
Easy Mailer 40-column	$69.96/$49.95
Easy Writer 40-column	$99.95/$74.95
Pro Easy Mailer	$175.00/$124.95
Pro Easy Writer	$249.95/179.95
Super Scribe II	$129.95/$79.95

We accept: MASTERCARD, VISA (Include card # and expiration date). CASHIER or CERTIFIED CHECKS, MONEY ORDERS, or PERSONAL CHECKS (Please allow 10 days to clear).
Please add 3% for shipping & handling (minimum $2.00) Foreign orders please add 10% for shipping & handling.
We accept COD's (Please Include $2.00 COD charge). California residents add 6% sales tax. All equipment is subject to price change and availability without notice All equipment is new & complete with manufacturer's warranty.

Apple is a registered trademark of APPLE COMPUTER INC

YOUR SALVATION IN THE SEA OF INFLATION

ARK COMPUTING

P.O. BOX 2025
CORONA, CALIFORNIA 91720
(714) 735-2250

Auto SAVE for the PET

by Louis F. Sander

This routine makes automatic cassette backup copies of your PET BASIC programs at regular intervals. Properly used, it can keep you from losing your temper and your creative programming work.

AUTOSAVE
requires:
PET with cassette

This short subroutine averts the anguish, anxiety, and anger that result from system crashes during BASIC program development. AUTOSAVE makes timely backup copies of the work in progress. So if you've ever seen your best creative effort swallowed up by a machine whose keyboard is locked out, read on for help.

You use the AUTOSAVE subroutine by putting it at the very end of your program, and by putting a GOSUB call at the very beginning. Then each time you RUN your main program, which you will certainly do as you check the lines you are writing, AUTOSAVE will be called first. If an hour or half hour point has passed since the last SAVE, AUTOSAVE will ask you if you'd like to make some backup copies. If you would, AUTOSAVE will make as many as you wish, using a 16-character program name consisting of the date and time of the SAVE, plus an eight-character identifier chosen earlier by you. AUTOSAVE will also give you special guidance in handling and marking your backup cassette.

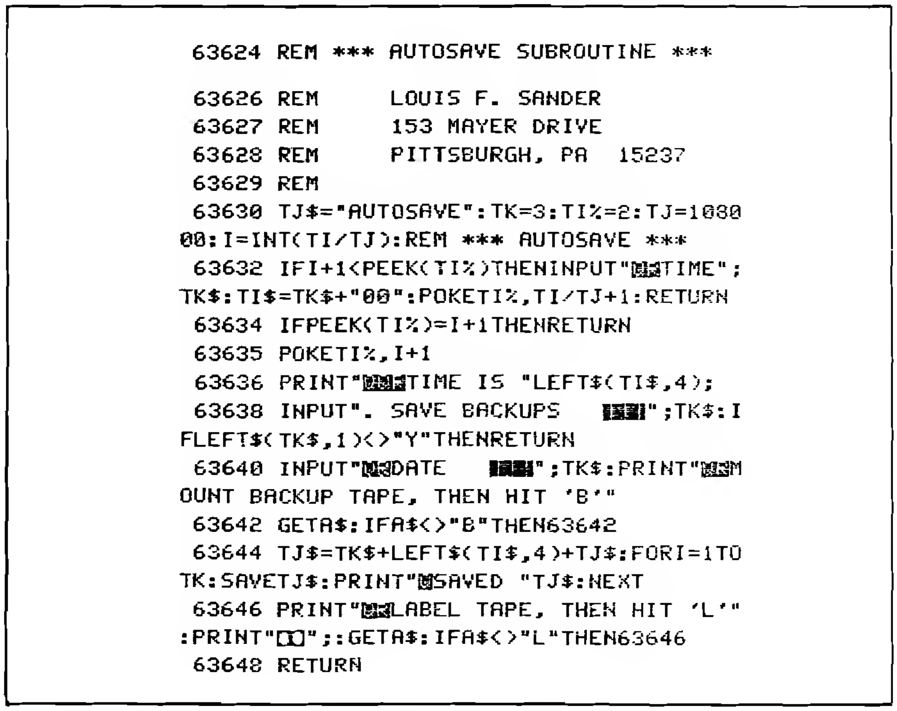
Then, if disaster strikes, just load your most recent backup and repeat your work since the time of its SAVE. In the worst case, you might lose 30 minutes worth of work. Since each backup's name includes the time it was saved, the most recent one will be easy to find.

Key AUTOSAVE into your PET exactly as it is listed, then save it on tape for further use. It is important that lines 63630 to 63648 take up no more than 17 screen lines, so don't use any unnecessary spaces in them. The six characters immediately after 'BACKUPS' in line 63638 and 'DATE' in line 63640, consist of three SHIFTED SPACEs, followed by three CURSOR LEFTs. Shifting the spaces is very important because it lets you answer an INPUT statement by pressing RETURN.

Line 63630 requires a bit of explanation. You will normally change this line, making TJ$ = the eight-character name of your main program. TK is the number of saves to be made at backup time, which you might also want to change. TI% is a POKE location used by the subroutine, and TI% = 2 is fine unless your main program calls the rarely used USR function, which also uses location 2. Possible (but untested) alternates in this case are TI% = 6 for original ROMs and TI% = 15 for all others.

If you are starting your main program from scratch, just load AUTOSAVE before you begin; if the main program is already on tape, add AUTOSAVE with an APPEND program. Or, just follow these steps:

1 . LOAD and LIST AUTOSAVE. Don't touch the keyboard after the LIST.
2. Mount the main program tape in TAPE #1, so that the main program is the next one on the tape.
3. Type LOAD, but *don't hit RETURN.*
4. Press PLAY, *then* quickly hit RETURN.



```
 63624 REM *** AUTOSAVE SUBROUTINE ***

 63626 REM     LOUIS F. SANDER
 63627 REM     153 MAYER DRIVE
 63628 REM     PITTSBURGH, PA  15237
 63629 REM
 63630 TJ$="AUTOSAVE":TK=3:TI%=2:TJ=1080
00:I=INT(TI/TJ):REM *** AUTOSAVE ***
 63632 IFI+1<PEEK(TI%)THENINPUT"▓▓TIME";
TK$:TI$=TK$+"00":POKETI%,TI/TJ+1:RETURN
 63634 IFPEEK(TI%)=I+1THENRETURN
 63635 POKETI%,I+1
 63636 PRINT"▓▓▓TIME IS "LEFT$(TI$,4);
 63638 INPUT". SAVE BACKUPS   ▓▓▓";TK$:I
FLEFT$(TK$,1)<>"Y"THENRETURN
 63640 INPUT"▓▓DATE   ▓▓▓";TK$:PRINT"▓▓M
OUNT BACKUP TAPE, THEN HIT 'B'"
 63642 GETA$:IFA$<>"B"THEN63642
 63644 TJ$=TK$+LEFT$(TI$,4)+TJ$:FORI=1TO
TK:SAVETJ$:PRINT"▓SAVED "TJ$:NEXT
 63646 PRINT"▓▓LABEL TAPE, THEN HIT 'L'"
:PRINT"□□";:GETA$:IFA$<>"L"THEN63646
 63648 RETURN
```

5. When the main program has loaded, AUTOSAVE's line 63630 should be at the top of your screen. HOME your cursor and add the AUTOSAVE lines to your program by hitting RETURN eleven times.

Now edit line 63630 so that TJ% = the name of your main program, and add a GOSUB 63630 as the first line of your main program. If for some reason you don't expect to RUN your main program from the beginning much while writing code, put some GOSUB 63630's in the program sections you will be running. Also finish the main program with an END statement to keep it from falling through to AUTOSAVE.

Here are some final notes: Entering a GOSUB 63636 in immediate mode will generate an automatic save at any time. Notice that 63636 is a symmetrical number, and therefore easy to remember. When AUTOSAVE asks you for the time, it will only accept 4-digit numbers from 0000 to 2359. It will accept anything at all as a date, but it's really looking for four digits (e.g., 0101 for New Year's Day). AUTOSAVE's special tape handling instructions are designed to have you take the cassette out of the recorder and label it; both steps inspire good operating practice. AUTOSAVE has been thoroughly tested on an original ROM PET. Since it depends on TI$ for its success, if your main program alters TI$ or shuts off the jiffy clock, AUTOSAVE will not work properly.

If you'd like a detailed description of the theory behind AUTOSAVE, just drop me a line at the address provided.

Louis F. Sander designs and markets electronic systems for hospitals and other health care providers. He is the originator of COMPUTER KINDERGARTEN™, a computer familiarization course for adults, and has written computer-related articles for several publications. He has worked with electronics since pre-transistor days. You may contact Mr. Sander at 153 Mayer Drive, Pittsburgh, Pennsylvania 15237.

MICRO

The chase is on, and you're *it!*

There are secret hordes of cash hidden in the city and you intend to "liberate" it all. But, someone tipped off the killer goons and they're hot on your tail. Floor that pedal . . . burn up the road!

Oh, oh . . . that's not enough to shake them. You need exceptional skill and strategy to lose them in the intricate and crazy maze of streets . . . to force them to crash, to trap them in dead ends.

They're **dead** serious – so don't relax. Get the cash and head out! Where? To another magnificent crazy maze of streets and smarter enemies. The better you are the more loot you grab, and the more you fight – up to 19 killer cars at a time!

Here's the most intricate hi-res mazes you've ever seen – to challenge your skill, agility and thinking ability. Get with CRAZY MAZEY now and run for your life!

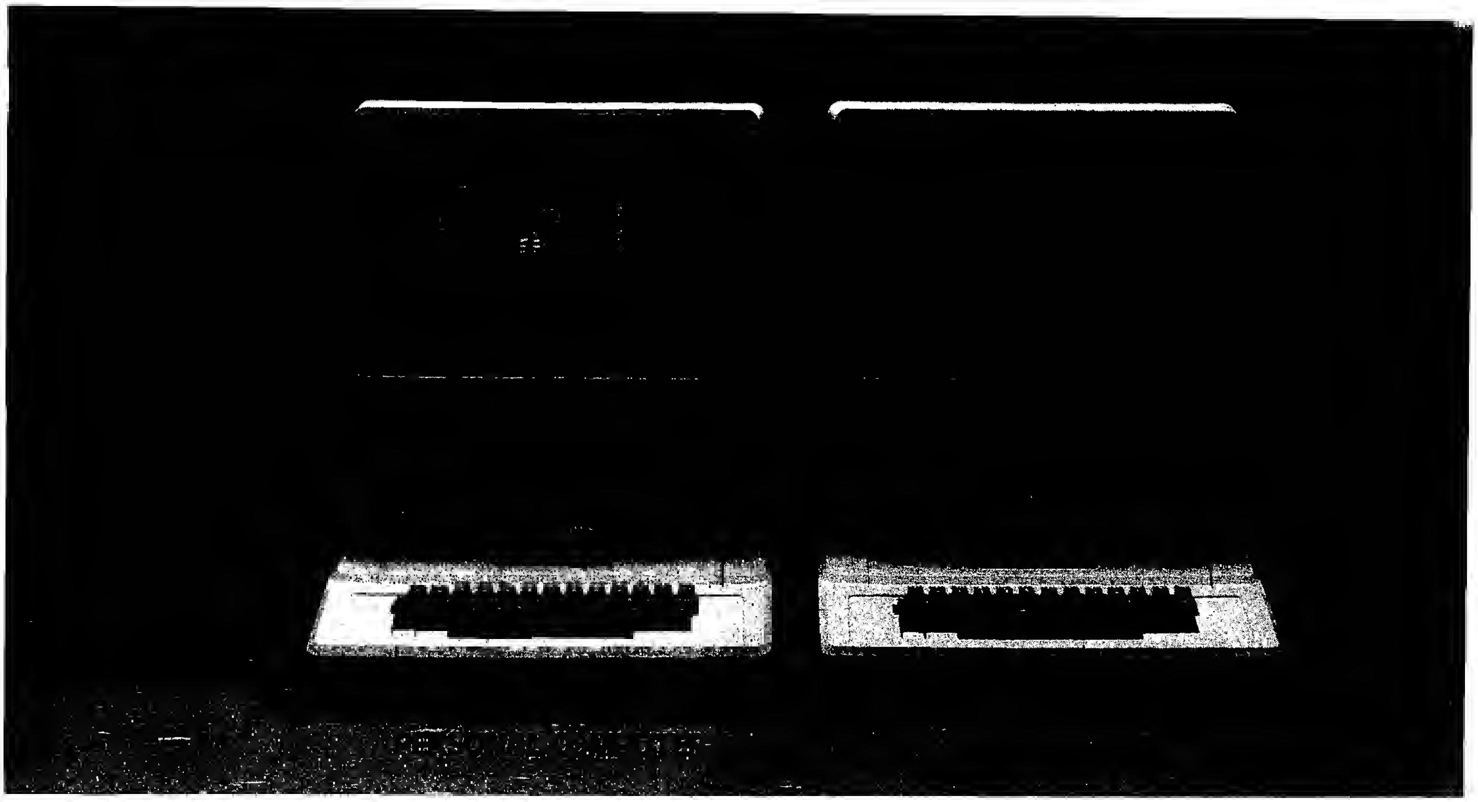
$29.95 For Apple II.* At you computer store, or from:

DATAMOST

9748 Cozycroft Ave., Chatsworth, CA 91311 (213) 709-1202

VISA/MASTERCHARGE accepted. $1.00 shipping/handling charge.
(California residents add 6½% sales tax.)

*Apple II is a trademark of Apple Computer, Inc.



TURN YOUR APPLE™ PERSONAL COMPUTER INTO A PROFESSIONAL COMPUTER FOR $750.

The majority of all professional computer software programs available today are written for the CP/M® disk operating system. The SYNERGIZER lets you access all of this vast body of sophisticated software with your Apple II while retaining the capability to access your present Apple software.

In addition to the CP/M interface and software diskette, the SYNERGIZER gives you the required 80 column display and 16K RAM memory expansion boards, the *CP/M Handbook* by Rodnay Zaks, and complete manuals. You get *everything* you need for fast, easy installation and operation in one package.* Each element is designed to complement the others, and everything is designed and produced by the *same* company.

The SYNERGIZER. It'll turn your Apple into a professional computer. And it costs only $750. Phone or write us, or ask your dealer for a SYNERGIZER brochure. Now.

SYNERGIZER™

Manufactured by Advanced Logic Systems,
1195 East Arques Avenue, Sunnyvale, CA 94086, (800) 538-8177 (In California (408) 730-0306)

Apple and Apple II are registered trademarks of Apple Computer, Inc.,
CP/M is a registered trademark of Digital Research, Inc.,
The CP/M Handbook is copyrighted by Sybex, Inc
And the SYNERGIZER was *our* idea.
*All SYNERGIZER Components are also sold separately.

TM

New Publications

Implementing BASICs, by William Payne and Patricia Payne. Reston Publishing Co., Inc. (11480 Sunset Hills Rd., Reston, VA 22090) 1982, 210 pages, 9¼ × 6 inches, hard cover.
ISBN: 0-8359-3045-9

This book is designed to help you achieve better software system designs and more programming techniques.

CONTENTS: Language Commands, Statements, and Their Variables; Microcomputer Data Structures; Variable Table Structure; Common Variables; Lexical Analysis, Text Atomization, and Syntax Analysis; Program Resolution; Program Text Coordinates; Interpreted Program Execution; Compiled BASICs; Verb Failures, User-Defined Verbs, and BASIC Line Editor; Timesharing Language Systems; Language System Code and Its Systems Verbs; How to Write a Language System; Conclusions and References; Appendix; Annotated Glossary of Techical Terms; Index.

Apple Pascal: A Hands-on Approach, by Arthur Luehrmann and Herbert Peckham. McGraw-Hill Book Company (New York, NY), 1981, 430 pages, 6½ × 9 inches, wire-o binding, paperback.
ISBN: 0-07-049171-2 $14.95

A series of sessions (chapters) designed to teach you Pascal, in a method similar to the way you learned English. The 14 sessions provide 30 to 40 hours of hands-on activities for the Pascal novice.

CONTENTS: Preface; Introduction; Getting Started; Typing in Programs — The EDITOR; Writing, Running, and Changing Programs; Generating Sound; Inventing New Words: Procedures; More Invented Words: Functions; Drawing Pictures; Branching Statements: IF and CASE; String Variables and WHILE Loops; Number Types and Arithmetic; Scalar Data Types and Sets; Arrays; Records and Files; Recursion; Where Do You Go From Here? Appendixes; Solutions to Problems; Index; Compiler Error Messages.

Elementary BASIC: Learning to Program Your Computer in BASIC with Sherlock Holmes, by Henry Ledgard and Andrew Singer. Vintage Books (201 E. 50th St., NY, NY 10022), 1982, 264 pages, 9¼ × 6 inches, paperback.
ISBN: 0-394-70789-3 $12.95

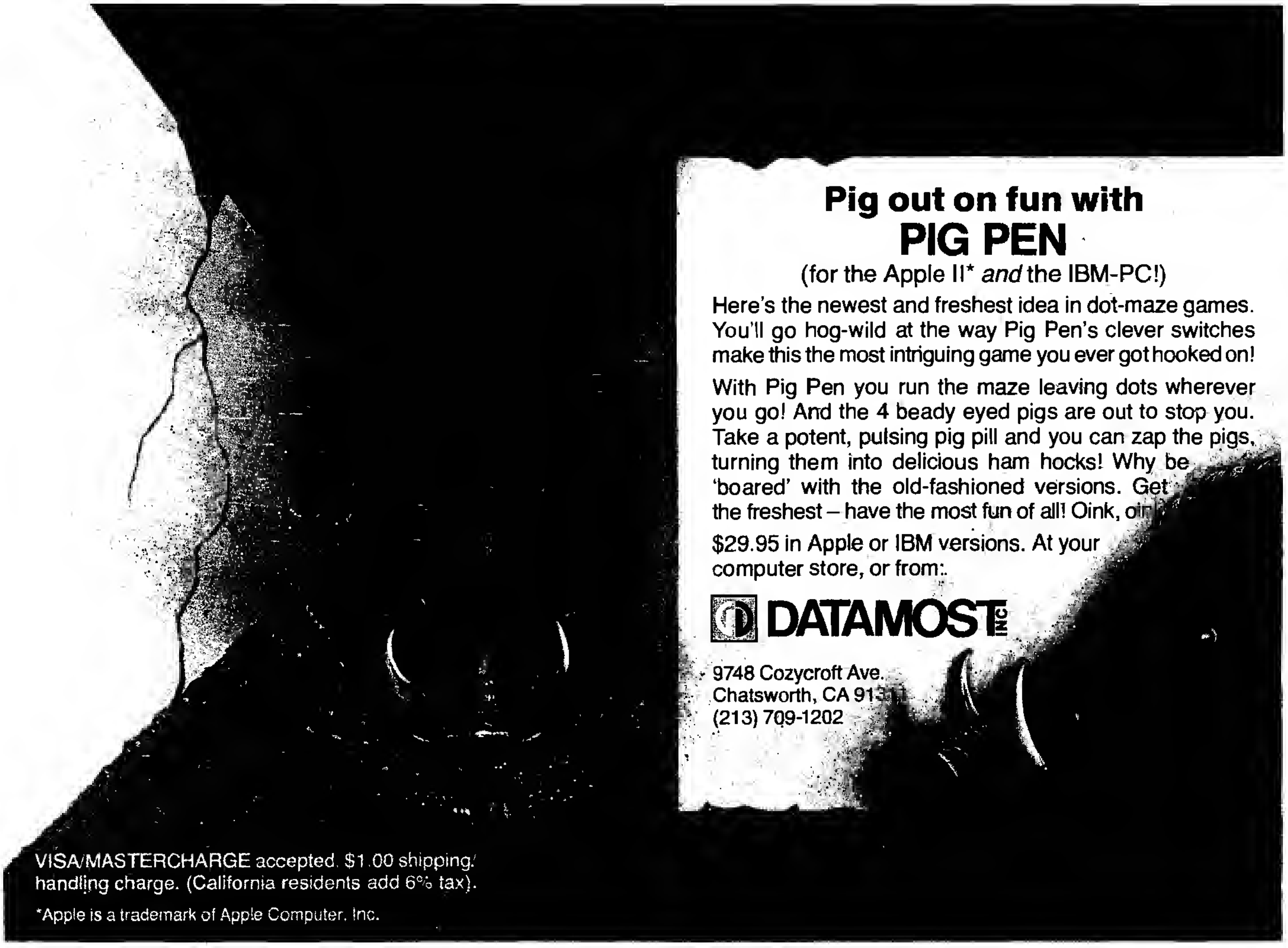
In these new Sherlock Holmes Stories, the great detective uses a computer to analyze clues. Holmes instructs Dr. Watson (and the reader) in a way that illuminates the mysteries of computer programming.

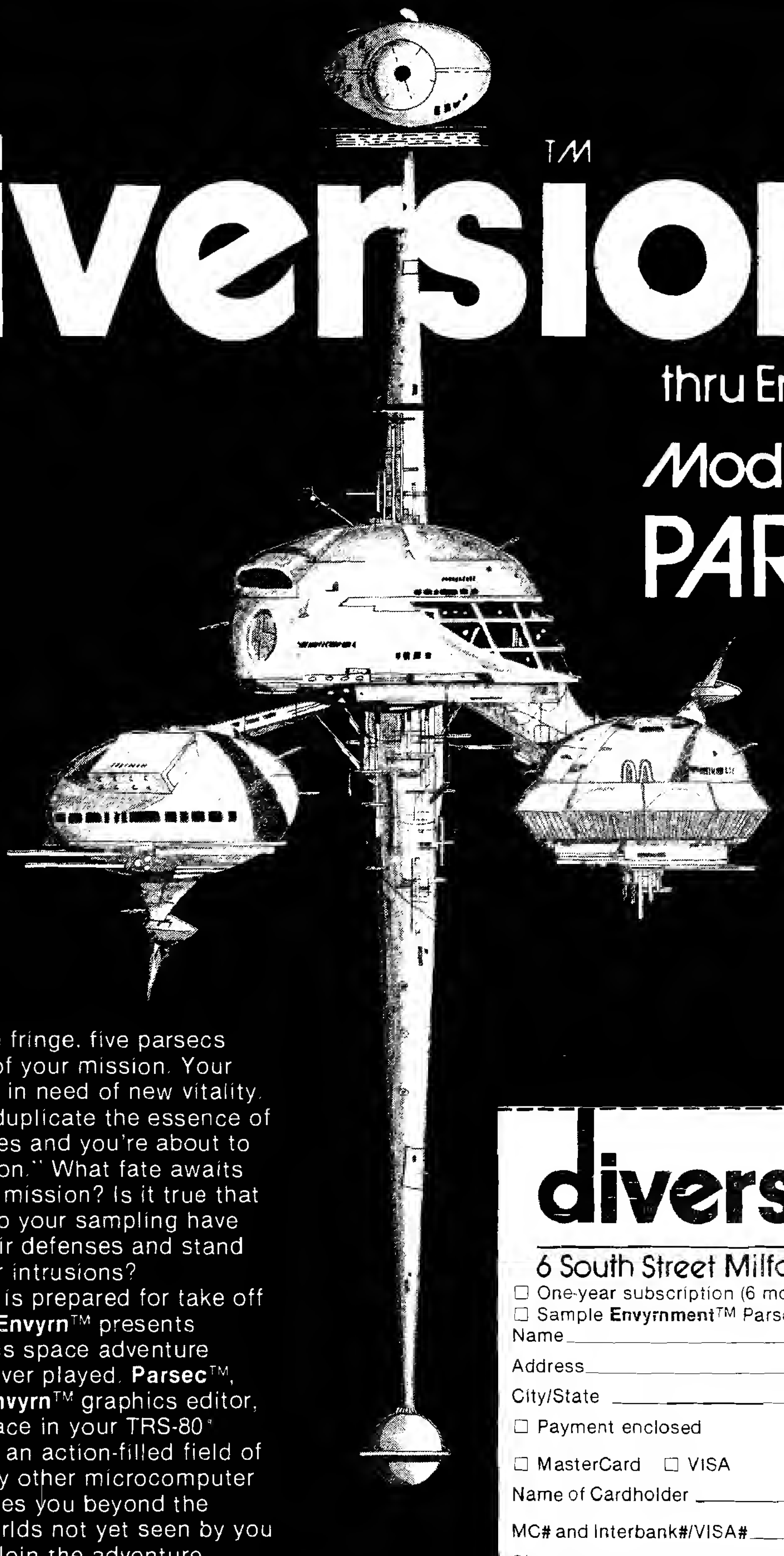
CONTENTS: The Analytical Engine; Murder at the Metropolitan Club; Holmes Gives a Demonstration; The Adventure of the Bathing Machine; A Study in Cigar Ash; The Adventure of Clergyman Peter; Holmes's Method Revealed; The Ciphered Message; An Advertisement in the *Times*; A Study in Chemistry; The Coroner's Report; The Adventure of the Gold Chip; Holmes Delivers a Lecture; The Final Programme; Appendix; Postscript; Index.

DP Directory, (P.O. Box 562, Bloomfield, CT 06002), monthly, 8½ × 11 inches, paperback.
ISSN: 0730-6806 $48.00/year

A collection of Table of Contents from DP magazines. Provides data processing managers, analysts and researchers with a tool to identify important articles. Over fifty magazines listed.

MICRO™





diversions™
thru Envyrn
Module One
PARSEC™

COMPRESS — An Applesoft Optimizer

by Barton M. Bauers

This machine-language utility will not only reduce Applesoft program memory requirements by up to 30% or more, but will increase execution and load speed also.

COMPRESS
requires:

Apple II or Apple II Plus
48K, Applesoft BASIC

Note: While the listing for COMPRESS is longer than we normally publish, MICRO feels this valuable program should be available to our readers. However, if you prefer not to key it in, MICRO will supply the program on DOS 3.3 disk for $10.00, plus $2.00 for shipping and handling. See page 113 for order information.

Recently announced Applesoft compilers are advertised to be up to twenty times faster than the Applesoft interpreter. Most reviewers have found, however, that in practice these compilers perform at about twice the operating speed of the interpreter. Recognizing that program compilation involves a significant amount of overhead, many programmers have avoided them or reserved them for specific applications. Since compilers do not dramatically increase speed, and since some programs require that variables and arrays be dimensioned while the program is being executed (a feature most compilers don't support), the Applesoft interpreter is often the better compromise.

COMPRESS is a machine-language program that will optimize most programs in less than ten seconds. To ensure clarity, COMPRESS allows the use of many REM statements in source programs. Variables can be dimensioned within programs, and can appear in the program before they are dimensioned (another feature the compilers can't

***Listing 1:* COMPRESS**

```
9000                ORG $9000
9000                OBJ $800
9000 A900           LDA #$00        ; SET HIMEM VALUE TO
9002 8573           STA $73         ; PROTECT COMPRESS
9004 A98D           LDA #$8D
9006 8574           STA $74
9008 4CD003         JMP $3D0        ; RETURN TO A/S
900B A900           LDA #$00        ; INIT POINTERS AND
900D AA             TAX             ; REGISTERS USED
900E A8             TAY             ; BY COMPRESS
900F 18             CLC
9010 8506           STA $06
9012 8508           STA $08
9014 8DD095         STA $95D0
9017 8DE095         STA $95E0
901A 85FB           STA $FB
901C A908           LDA #$08
901E 8507           STA $07
9020 A98D           LDA #$8D
9022 8509           STA $09
9024 A907           LDA #$07
9026 851F           STA $1F
9028 A901           LDA #$01
902A 85EF           STA $EF
902C A900           LDA #$00
902E 8DE195         STA $95E1
9031 8DE295         STA $95E2
9034 8DE395         STA $95E3
9037 A903           LDA #$3         ; PUT DEC-HEX CONVERSION
9039 8DE495         STA $95E4       ; CONSTANTS IN MEMORY
903C A927           LDA #$27        ; TO PERMIT A/S DECIMALIZED
903E 8DE595         STA $95E5       ; TOKENS TO BE CONVERTED
9041 A901           LDA #$1         ; TO HEX LINE NUMBERS
9043 8DD195         STA $95D1
9046 A90A           LDA #$A
9048 8DD295         STA $95D2
904B A964           LDA #$64
904D 8DD395         STA $95D3
9050 A9E8           LDA #$E8
9052 8DD495         STA $95D4
9055 A910           LDA #$10
9057 8DD595         STA $95D5
905A A200           LDX #$00        ; START ROUTINE TO
905C A9FF           LDA #$FF        ; BLANK TABLES
905E                PAG
905E 9D008D  CLR1   STA $8D00,X     ; BLANK PAGE $8D
9061 E0FF           CPX #$FF
9063 F004           BEQ CLR2
9065 E8             INX
9066 4C5E90         JMP CLR1
9069 A200    CLR2   LDX #$00
906B 9D008E  CLR3   STA $8E00,X     ; BLANK PAGE $8E
906E E0FF           CPX #$FF
9070 F004           BEQ CLR4
9072 E8             INX
9073 4C6B90         JMP CLR3
9076 A200    CLR4   LDX #$00
9078 9D008F  CLR5   STA $8F00,X     ; BLANK PAGE $8F
907B E0FF           CPX #$FF
907D F03E           BEQ NOW
907F E8             INX
9080 4C7890         JMP CLR5
9083 C8      INCY   INY             ; SUBROUTINE TO INCREMENT
9084 C000           CPY #$00        ; Y-REG AND CHECK TO
9086 D008           BNE GOBAK       ; SEE IF PAGE BOUNDARY
9088 86ED           STX $ED         ; CROSSED
908A A61F           LDX $1F
908C F600           INC $00,X
908E A6ED           LDX $ED
9090 60      GOBAK  RTS
9091 20DDFB  ERROR  JSR $FBDD       ; ERROR ROUTINE TO
9094 A9C5           LDA #$C5        ; PRINT ERROR CODE
9096 20F0FD         JSR $FDF0       ; # AND RETURN
9099 A9D2           LDA #$D2        ; TO A/S ABORTING
909B 20F0FD         JSR $FDF0       ; COMPRESS
```

(Continued)

match). But actual control during execution must pass to those statement numbers after they are dimensioned. So COMPRESS does not interfere with any normal interpreter capabilities.

Although COMPRESS can be thought of as a poor man's compiler, it is not a compiler at all. It merely reduces Applesoft programs to their minimum configuration.

COMPRESS occupies 2.3 kilobytes of free RAM memory, just below DOS in a 48K Apple II Plus. It also uses about 30 bytes of zero-page storage for pointers. Typical programs can be reduced 15% or more in memory requirements, with up to 30% or more for certain programs that contain a large number of REMs or other redundant code. But COMPRESS is not simply a REM remover. The programmer is encouraged to insert a great deal of narrative in the original source program so that any modifications required later can be easily made to the source code and then recompressed.

What COMPRESS Does

COMPRESS removes line numbers, redundant statements, etc., and concatenates as many Applesoft statements as can be legally assigned to one line number. For example, consider this program:

```
10 LET J = 5
20 LET K = 10
30 END
```

Although this simple program does nothing, it is a legal Applesoft program and could be run through COMPRESS. After running COMPRESS, the program would appear as:

```
10 J = 5 : K = 10 : END
```

COMPRESS has removed 45% of the memory requirements of this program!

The original program would have appeared in memory as:

```
00  0A  08  0A  00  AA  4A  D0
35  00  14  08  14  00  AA  4B
D0  31  30  00  1A  08  1E  00
80  00  00  00
```

The compressed program in memory would appear as:

```
00  10  08  0A  00  4A  D0  35
3A  4B  D0  31  30  3A  80  00
00  00
```

Note how many tokens are removed after the original program is cleaned up.

Listing 1 *(continued)*

```
909E 20F0FD          JSR $FDF0
90A1 A9CF            LDA #$CF
90A3 20F0FD          JSR $FDF0
90A6 A9D2            LDA #$D2
90A8 20F0FD          JSR $FDF0
90AB A9A0            LDA #$A0
90AD 20F0FD          JSR $FDF0
90B0 A9A3            LDA #$A3
90B2 20F0FD          JSR $FDF0
90B5 A5F9            LDA $F9
90B7 20DAFD          JSR $FDDA
90BA 4CD003          JMP $3D0
90BD B106    NOW     LDA ($06),Y         ; PROGRAM STARTS HERE
90BF 208390          JSR INCY            ; GET A BYTE - INC Y-REG
90C2 C900            CMP #$00            ; IS IT 0?
90C4 F03C            BEQ TSTR            ; YES - CHECK FOR END OF PROGRAM
90C6 C9C4            CMP #$C4            ; IS IT ´THEN´?
90C8 F013            BEQ THEN            ; YES
90CA C9AB    NTHG    CMP #$AB            ; IS IT ´GOTO´?
90CC                 PAG
90CC F055            BEQ LOAD            ; YES
90CE C9B0            CMP #$B0            ; IS IT ´GOSUB´?
90D0 F051            BEQ LOAD            ; YES
90D2 C9BC            CMP #$BC            ; IS IT ´LIST´?
90D4 F04D            BEQ LOAD            ; YES
90D6 C985            CMP #$85            ; IS IT ´DEL´?
90D8 F049            BEQ LOAD            ; YES
90DA 4CBD90          JMP NOW             ; GO GET ANOTHER BYTE
90DD 85FD    THEN    STA $FD             ; PUT A-REG ASIDE
90DF B106            LDA ($06),Y         ; GET ANOTHER BYTE
90E1 208390          JSR INCY            ; INC Y-REG
90E4 C9AB            CMP #$AB            ; IS IT ´GOTO´?
90E6 F03B            BEQ LOAD            ; YES
90E8 38              SEC                 ; FIND OUT IF THE TOKEN IS
90E9 E930            SBC #$30            ; A LINE NUMBER REF.
90EB 3005            BMI CLRIT           ; NO, IT ISIN´T
90ED 38              SEC
90EE E90A            SBC #$0A
90F0 3008            BMI GOTIT           ; YES IT IS!
90F2 208893  CLRIT   JSR DECY            ; BACK UP ONE
90F5 A5FD            LDA $FD             ; GET A-REG BACK
90F7 4CBD90          JMP NOW             ; FALSE ALARM
90FA 208893  GOTIT   JSR DECY            ; BACK UP ONE
90FD A5FD            LDA $FD             ; GET A-REG BACK
90FF 4C2391          JMP LOAD            ; GO FORWARD
9102 B106    TSTR    LDA ($06),Y         ; GET ANOTHER BYTE
9104 208390          JSR INCY            ; INC Y-REG
9107 C900            CMP #$00            ; IS THIS 0 ALSO?
9109 F00C            BEQ PEND            ; YES - 2 IN A ROW
910B 208390          JSR INCY            ; NO.INC Y-REG
910E 208390  NOTND   JSR INCY            ; INC Y REG AGAIN
9111 208390          JSR INCY            ; AND AGAIN
9114 4CBD90          JMP NOW             ; CONTINUE ON
9117 B106    PEND    LDA ($06),Y         ; LOOK AT NEXT BYTE
9119 208390          JSR INCY            ; INC Y-REG
911C C900            CMP #$00            ; IS THIS THE THIRD 0?
911E D0EE            BNE NOTND           ; NO-THE PROGRAM IS NOT ENDED
9120 4C2292          JMP QEND            ; YES - IT´S ALL OVER
9123 B106    LOAD    LDA ($06),Y         ; GET NEXT BYTE
9125 C900            CMP #$00            ; IS IT 0?
9127 F02E            BEQ NUMDUN          ; YES THE STATEMENT IS ENDED
9129 208390          JSR INCY            ; INC Y-REG
912C C93A            CMP #$3A            ; IS IT A ´COLON´?
912E F027            BEQ NUMDUN          ; YES THE STATEMENT IS ENDED
9130 C92C            CMP #$2C            ; IS IT A COMMA?
9132 F00D            BEQ COMMA           ; YES
9134 A6EF            LDX $EF             ; GET THE STATEMENT BUFFER OFFSET
9136 38              SEC
9137 E930            SBC #$30            ; DECMALIZE THE TOKEN

9139                 PAG
9139 9DF095          STA $95F0,X         ; PUT IT ASIDE
913C E6EF            INC $EF             ; ADD 1
913E 4C2391          JMP LOAD            ; GO GET ANOTHER BYTE
9141 C6EF    COMMA   DEC $EF             ; COMMA MEANS LINE NUMBER ENDED
9143 206391          JSR CONV            ; GO CONVERT STMT #
9146 A901            LDA #$01            ; REINIT
9148 85EF            STA $EF
914A B106            LDA ($06),Y         ; GET ANOTHER BYTE
914C C900            CMP #$00            ; IS IT 0?
914E F010            BEQ COMFIN          ; YES - LINE IS DONE
9150 C93A            CMP #$3A            ; IS IT COLON?
9152 F00C            BEQ COMFIN          ; YES - LINE IS DONE
9154 4C2391          JMP LOAD            ; NO - CONTINUE ON
9157 C6EF    NUMDUN  DEC $EF             ; ALL OF # IS FOUND
9159 206391          JSR CONV            ; GO CONVERT STMT #
915C A901            LDA #$01            ; REINIT
915E 85EF            STA $EF
9160 4CBD90  COMFIN  JMP NOW             ; BEGIN SEARCH OF NEXT LINE
9163 84FC    CONV    STY $FC             ; PUT Y-REG ASIDE
9165 A6EF            LDX $EF             ; GET $95F0 OFFSET
9167 86FF            STX $FF             ; SAVE IT
9169 A900            LDA #$00
916B 85FE            STA $FE
916D 85FA            STA $FA
916F A6FF    GO      LDX $FF
9171 BCF095          LDY $95F0,X         ; GET FIRST DEC DIGIT
9174 C6FF            DEC $FF             ; DECREMENT OFFSET
9176 E6FE            INC $FE             ; INCREMENT COUNTER
9178 C000            CPY #$00            ; IS THE DEC # 0?
```

Listing 1 *(continued)*

```
917A F019                BEQ NEXT       ; YES - GET NEXT
917C A6FE                LDX $FE        ; NO - INC COUNTER
917E BDD095              LDA $95D0,X    ; GET APPROPRIATE DEC/HEX CONSTANT
9181 88         CONTA    DEY            ; COUNT DOWN 1
9182 C000                CPY #$00       ; IS Y 0?
9184 F00F                BEQ NEXT       ; YES - GET NEXT
9186 18                  CLC            ; NO - CONVERT
9187 7DD095              ADC $95D0,X    ; CONVERSION OF CONSTANT
918A B003                BCS INCR       ; IF OVERFLOW BRANCH
918C 4C8191              JMP CONTA      ; CONTINUE
918F EEE095     INCR     INC $95E0      ; ADD 1 TO HEX MSB
9192 4C8191              JMP CONTA      ; CONTINUE
9195 18         NEXT     CLC            ; GET NEXT DIGIT
9196 6DD095              ADC $95D0
9199 8DD095              STA $95D0      ; STORE RESULT
919C B028                BCS INCS       ; IF OVERFLOW BRANCH
919E A6EF       CONTC    LDX $EF        ; GET COUNTER
91A0 BCF095              LDY $95F0,X    ; GET NEXT CONSTANT
91A3 C6EF                DEC $EF        ; DECREMENT COUNTER
91A5 E6FA                INC $FA        ; INCREMENT POINTER
91A7                     PAG
91A7 C000                CPY #$00       ; IS Y 0?
91A9 F034                BEQ OUT        ; YES - QUIT
91AB A6FA                LDX $FA        ; NO - GET POINTER
91AD BDE095              LDA $95E0,X    ; GET DEC -> HEX CONSTANT
91B0 88         CONTB    DEY            ; DECREMENT COUNTER
91B1 C000                CPY #$00       ; IS Y 0?
91B3 F017                BEQ ADDM       ; YES - EXIT
91B5 18                  CLC            ; NO - CONVERT
91B6 7DE095              ADC $95E0,X
91B9 B003                BCS INCA       ; BRANCH IF OVERFLOW
91BB 4CB091              JMP CONTB      ; CONTINUE
91BE 18         INCA     CLC
91BF A902                LDA #$02       ; GET ERROR CODE
91C1 85F9                STA $F9        ; HEX > $FFFF
91C3 4C9190              JMP ERROR      ; GO PRINT ERROR MSG
91C6 EEE095     INCS     INC $95E0      ; ADD $1 TO MSB
91C9 4C9E91              JMP CONTC      ; CONTINUE
91CC 18         ADDM     CLC
91CD 6DE095              ADC $95E0
91D0 B006                BCS INCB       ; BRANCH IF OVERFLOW
91D2 8DE095              STA $95E0      ; STORE RESULT
91D5 4CDF91              JMP OUT        ; CONTINUE
91D8 A901       INCB     LDA #$01       ; ERROR #1
91DA 85F9                STA $F9
91DC 4C9190              JMP ERROR      ; GO PRINT ERROR MSG
91DF A5FF       OUT      LDA $FF        ; ARE WE DONE?
91E1 C900                CMP #$00
91E3 F005                BEQ WRITE      ; YES -
91E5 A900                LDA #$00       ; NO
91E7 4C6F91              JMP GO         ; GO ON
91EA A909       WRITE    LDA #$09       ; SET UP POINTERS + OFFSETS
91EC 851F                STA $1F
91EE A4FB                LDY $FB
91F0 ADD095              LDA $95D0      ; GET LSB OF HEX LINENUM
91F3 9108                STA ($08),Y    ; STORE HEX VALUE
91F5 208390              JSR INCY       ; INCREMENT Y
91F8 ADE095              LDA $95E0      ; GET MSB OF HEX LINENUM
91FB 9108                STA ($08),Y    ; INCREMENT Y
91FD A509                LDA $09        ; LOOK AT MSB OF ADDRESS
91FF C98E                CMP #$8E       ; CHECK IF ROOM LEFT
9201 D00B                BNE OKNOW      ; YES
9203 C0FF                CPY #$FF       ; CHECK IF ROOM LEFT
9205 D007                BNE OKNOW      ; YES
9207 A903                LDA #$03       ; ERROR CODE
9209 85F9                STA $F9
920B 4C9190              JMP ERROR      ; GO PRINT ERROR MSG
920E 208390     OKNOW    JSR INCY       ; RESET POINTERS
9211 84FB                STY $FB
9213 A907                LDA #$07
9215                     PAG
9215 851F                STA $1F
9217 A4FC                LDY $FC
9219 A900                LDA #$00       ; GET ZERO
921B 8DD095              STA $95D0      ; STORE IN LSB REGISTER
921E 8DE095              STA $95E0      ; STORE IN MSB REGISTER
9221 60                  RTS            ; RETURN
9222 84FC       QEND     STY $FC        ; END OF TABLECREATE
9224 A9C5                LDA #$C5       ; PRINT MESSAGE TO SCREEN
9226 20F0FD              JSR $FDF0
9229 A9CE                LDA #$CE
922B 20F0FD              JSR $FDF0
922E A9C4                LDA #$C4
9230 20F0FD              JSR $FDF0
9233 A9A0                LDA #$A0
9235 20F0FD              JSR $FDF0
9238 A9CF                LDA #$CF
923A 20F0FD              JSR $FDF0
923D A9C6                LDA #$C6
923F 20F0FD              JSR $FDF0
9242 A9A0                LDA #$A0
9244 20F0FD              JSR $FDF0
9247 A9D0                LDA #$D0
9249 20F0FD              JSR $FDF0
924C A9C1                LDA #$C1
924E 20F0FD              JSR $FDF0
9251 A9D3                LDA #$D3
9253 20F0FD              JSR $FDF0
9256 20F0FD              JSR $FDF0
```

(Continued)

This example illustrates the basic action of the program COMPRESS.

Why COMPRESS Works

Applesoft programs are models of efficiency. There is no extraneous space or character storage in memory. Each command, character, or number is represented within the computer by a token. The token for the command PRINT is "BA", a hexadecimal number equivalent to the decimal value 186. There are 234 different tokens that Applesoft can recognize, represented by hexadecimal numbers $00 through $EA. (Following the normal convention, we will refer to all hexadecimal numbers with a prefix of '$' so they can be distinguished from normal decimal numbers.)

Continuing with the example PRINT ($BA), use of the token allows storage of the command PRINT in just one byte, rather than spread out over five. In effect, a great deal of optimizing has already been done in the design of the Apple!

Other computers carry blanks into the program memory, but the Apple only does this with strings, where each blank is significant. The token representation for the words "CUSTOMER NAME" would be:

43 55 53 54 4F 4D 45 52 20 4E 41 4D 45

In this case, since the programmer desires the string typed out exactly as entered, no changes would be acceptable within it, and Applesoft makes none.

Despite the fact that the Applesoft programs are already condensed in memory, additional optimizing is possible. Between each line in memory, for example, the computer stores the value $00. This null character, called a delimiter, tells the Applesoft interpreter that the previous line has ended. After $00, two bytes indicate the address in memory of the statement following the next statement. These bytes ($0A $08 in the example at the beginning) are stored in low-byte/high-byte format. The actual address in this example is therefore $080A.

The line number is stored in two bytes, again with the least significant byte first. Ever wonder why the largest line number which can be used in Applesoft is 63999? One reason is that two bytes can describe a maximum of $FFFF integers — the equivalent of decimal 65535.

Each Applesoft statement, then, requires five bytes — a null byte, two address bytes, and two line-number bytes.

As noted in the Applesoft manual, it is possible to put two or more statements together by separating each with a colon (:). If two statements are thus joined, only one byte is used ($3A, the token for the colon). Separate lines would require four additional bytes. Applesoft imposes only the limitation that no line can have more than 255 characters.

Applesoft BASIC contains other opportunities for optimization. For example, COMPRESS eliminates the unnecessary LET, thus saving one byte. And COMPRESS will delete any variable name after NEXT. This could save many bytes per line. REM statements can also be dropped, unless of course some branching statement jumps to a line number that happens to be a REM statement (a bad programming habit).

Even though Applesoft is a very efficient language, there is ample opportunity for further reduction in memory use when storing programs.

How COMPRESS Works

The first step in compressing a program is to look for called line numbers. COMPRESS will eliminate lines by appending them to earlier lines. But it is important to preserve lines that are referred to elsewhere in the program with commands such as GOTO, GOSUB, etc., to permit proper transfer of control. Therefore, the first action COMPRESS takes is the creation of a Called Line Number Table.

The following Applesoft commands refer control to another line number:

IF ... GOTO
GOTO
GOSUB
LIST
ON ... GOTO
DEL

COMPRESS searches the program for these commands. Line numbers that are identified in these statements are stored in a Called Line Number Table at memory locations $8D00 to $8EFF. When compression takes place, this table is checked so no line number that appears in the table as a called line number will be eliminated.

The table will contain all of the called line numbers in the order in which they appeared in the original program.

Listing 1

```
9259 A9A0           LDA #$A0
925B 20F0FD         JSR $FDF0
925E A9A3           LDA #$A3
9260 20F0FD         JSR $FDF0
9263 A9B1           LDA #$B1
9265 20F0FD         JSR $FDF0
9268 208BFD         JSR $FD8B
926B A900    PGMWRT LDA #$00        ; INITIALIZE VALUES FOR
926D A8             TAY             ; POINTERS AND OFFSETS
926E 8506           STA $06         ; FOR COMPRESSION
9270 85FF           STA $FF         ; PORTION OF PROGRAM
9272 85EB           STA $EB
9274 85EF           STA $EF
9276 85D6           STA $D6
9278 85CE           STA $CE
927A 8508           STA $08
927C 8DFE8F         STA $8FFE
927F A93A           LDA #$3A
9281 8DFF8F         STA $8FFF
9284 A908           LDA #$08
9286 8507           STA $07
9288                PAG
9288 85EC           STA $EC
928A A907           LDA #$07
928C 851F           STA $1F
928E A98F           LDA #$8F
9290 85CF           STA $CF
9292 A903           LDA #$03
9294 85FA           STA $FA
9296 B106    CMPPB  LDA ($06),Y     ; READ BYTE OF A/S PROG
9298 208390         JSR INCY
929B C900           CMP #$00        ; IS IT 0?
929D F003           BEQ TSTS        ; YES - TEST IT
929F 4C9E93         JMP PROG        ; NO - WRITE IT
92A2 A6FA    TSTS   LDX $FA
92A4 E003           CPX #$03        ; DID COMPRESS JUST START?
92A6 D00C           BNE TSTY        ; NO
92A8 8419           STY $19         ; YES - STORE LOCATION OF
92AA A4EC           LDY $EC         ; A/S ADDRESS REFERENCE
92AC 841A           STY $1A         ; IN COMPRESSED STATEMENT
92AE A419           LDY $19
92B0 A200           LDX #$00        ; RESET INDICATOR
92B2 86FA           STX $FA
92B4 841B    TSTY   STY $1B
92B6 C61B           DEC $1B
92B8 B106           LDA ($06),Y     ; GET ANOTHER BYTE
92BA 208390         JSR INCY
92BD C900           CMP #$00        ; IS IT 0 ALSO?
92BF D003           BNE GOON        ; NO - GO ON
92C1 4C9293         JMP REND        ; YES - CHECK FOR END
92C4 208390  GOON   JSR INCY
92C7 B106           LDA ($06),Y     ; GET ANOTHER BYTE
92C9 208390         JSR INCY
92CC A6FF           LDX $FF
92CE E000           CPX #$00        ; IS THIS NEW LINE?
92D0 F003           BEQ STRA        ; YES GO STORE
92D2 4CD792         JMP NSTR        ; NO
92D5 851D    STRA   STA $1D         ; SAVE LINE NUMBER
92D7 84FD    NSTR   STY $FD         ; PUT ASIDE POINTERS
92D9 A08D           LDY #$8D        ; TO CHECK CALLED LINE
92DB 84D7           STY $D7         ; NUMBER TABLE!
92DD A2D7           LDX #$D7
92DF 861F           STX $1F
92E1 A000           LDY #$00
92E3 C4FB    ANSTR  CPY $FB         ; ARE WE DONE CHECKING?
92E5 F011           BEQ ENDTST      ; YES
92E7 D1D6    NOMAT  CMP ($D6),Y     ; LOOK FOR LSB OF LINE #
92E9 F030           BEQ STRB        ; FOUND IT!
92EB 208390         JSR INCY
92EE C4FB           CPY $FB         ; ARE WE DONE?
92F0 F006           BEQ ENDTST      ; YES
92F2                PAG
92F2 208390         JSR INCY
92F5 4CE392         JMP ANSTR       ; NO - CONTINUE CHECKING
92F8 A609    ENDTST LDX $09
92FA E4D7           CPX $D7         ; CHECK LIMITS OF CALLED
92FC D0E9           BNE NOMAT       ; LINE # TABLE
92FE A4FD           LDY $FD
9300 A207           LDX #$07
9302 861F           STX $1F
9304 B106           LDA ($06),Y     ; GET ANOTHER A/S BYTE
9306 208390         JSR INCY
9309 A6FF           LDX $FF
930B E000           CPX #$00        ; IS INDICATOR CLEAR?
930D F003           BEQ STAE        ; YES
930F 4C9692         JMP CMPPB       ; NO - GET A BYTE
9312 851E    STAE   STA $1E         ; STORE MSB OF LINE #
9314 A904           LDA #$04        ; CHANGE POINTER
9316 85FA           STA $FA
9318 4C9692         JMP CMPPB       ; GO GET NEXT BYTE
9318 208390  STRB   JSR INCY
931E 84FE           STY $FE         ; SET UP POINTERS FOR
9320 A4FD           LDY $FD         ; ADDITIONAL CHECKING OF
9322 A207           LDX #$07        ; MSB OF LINE NUMBER
9324 861F           STX $1F
9326 851C           STA $1C
9328 B106           LDA ($06),Y     ; GET A BYTE OF A/S
932A 208390         JSR INCY
932D A6FF           LDX $FF
```

Listing 1 *(continued)*

```
932F E000           CPX #$00          ; IS INDICATOR CLEAR?
9331 F003           BEQ STAF          ; YES -
9333 4C3893         JMP CNTXX         ; NO
9336 851E    STAF   STA $1E           ; PUT MSB AWAY
9338 84FD    CNTXX  STY $FD           ; START LOOKING FOR
933A A2D7           LDX #$D7          ; A MATCH OF THE MSB
933C 861F           STX $1F           ; OF THE LINE NUMBER
933E A4FE           LDY $FE
9340 D1D6           CMP ($D6),Y
9342 F01B           BEQ SET
9344 208390         JSR INCY
9347 84ED           STY $ED
9349 A007           LDY #$07
934B 841F           STY $1F
934D A4FD           LDY $FD
934F 208893         JSR DECY
9352 84FD           STY $FD
9354 A0D7           LDY #$D7
9356 841F           STY $1F
9358 A4ED           LDY $ED
935A A51C           LDA $1C
935C 4CE392         JMP ANSTR

935F                PAG
935F A4FD    SET    LDY $FD           ; RETURN WITH A MATCH
9361 A207           LDX #$07
9363 861F           STX $1F
9365 A205           LDX #$05
9367 86FA           STX $FA           ; RESET FLAG
9369 A6FF           LDX $FF           ; IS THIS A NEW LINE?
936B E000           CPX #$00
936D D003           BNE GOGO          ; NO
936F 4C9692         JMP CMPPB         ; YES - GET ANOTHER BYTE
9372 A901    GOGO   LDA #$01          ; RESET FLAG
9374 85FA           STA $FA
9376 208893         JSR DECY          ; BACK UP POINTER
9379 208893         JSR DECY
937C 208893         JSR DECY
937F 208893         JSR DECY
9382 208893         JSR DECY
9385 4CD894         JMP CERTA         ; MATCH MADE - GO WRITE LINE
9388 88      DECY   DEY               ; SUBROUTINE TO DECREMENT
9389 C0FF           CPY #$FF          ; Y REG AND CHECK IF PAGE
938B D004           BNE RETNN         ; BOUNDARY IS CROSSED
938D A61F           LDX $1F
938F D600           DEC $00,X
9391 60      RETNN  RTS
9392 B106    REND   LDA ($06),Y       ; GET ANOTHER BYTE
9394 C900           CMP #$00          ; IS THIS ONE 0 ALSO?
9396 D003           BNE GONO          ; NO - IT IS NOT THE END
9398 4C6895         JMP FEND          ; YES - PROGRAM ENDED
939B 4CC492  GONO   JMP GOON          ; GO ON
939E 8DFE8F  PROG   STA $8FFE         ; CHECK TO SEE IF WE
93A1 C93A           CMP #$3A          ; HAVE A COLON
93A3 F00D           BEQ FLAG          ; YES
93A5 ADFF8F         LDA $8FFF         ; NO-CHECK THIS ADDRESS
93A8 C93A           CMP #$3A
93AA F02C           BEQ CLRFLG
93AC ADFE8F         LDA $8FFE
93AF 4CE093         JMP ONWARD
93B2 ADFF8F  FLAG   LDA $8FFF         ; CHECK TO SEE
93B5 C93A           CMP #$3A          ; IF WE HAVE A COLON
93B7 D00D           BNE SETZ          ; NO
93B9 B106           LDA ($06),Y       ; YES GET ANOTHER BYTE
93BB C900           CMP #$00          ; IS IT 0?
93BD F003           BEQ RNGFIX        ; YES
93BF 4C9692         JMP CMPPB         ; NO GO CONTINUE
93C2 CA      RNGFIX DEX               ; BACK UP ONE
93C3 4C9494         JMP CNTAX
93C6 B106    SETZ   LDA ($06),Y       ; GET NEXT BYTE
93C8 C900           CMP #$00          ; IS IT 0?
93CA D003           BNE SETZ1         ; NO
93CC 4C9494         JMP CNTAX         ; YES
93CF                PAG
93CF ADFE8F  SETZ1  LDA $8FFE         ; SET FLAG
93D2 8DFF8F         STA $8FFF
93D5 4CF893         JMP PROGWT        ; WRITE COMPRESSED CODE
93D8 A900    CLRFLG LDA #$00
93DA 8DFF8F         STA $8FFF
93DD ADFE8F         LDA $8FFE
93E0 C9AB    ONWARD CMP #$AB          ; IS IT 'GOTO'?
93E2 F039           BEQ GOTO          ; YES
93E4 C9AD           CMP #$AD          ; IS IT 'IF'?
93E6 F050           BEQ IF            ; YES
93E8 C9B1           CMP #$B1          ; IS IT 'RETURN'?
93EA F04C           BEQ IF            ; YES
93EC C982           CMP #$82          ; IS IT 'NEXT'?
93EE F053           BEQ NXTT          ; YES
93F0 C9AA           CMP #$AA          ; IS IT 'LET'?
93F2 F066           BEQ LET           ; YES
93F4 C9B2           CMP #$B2          ; IS IT 'REM'?
93F6 F06A           BEQ REM           ; YES
93F8 9D008F  PROGWT STA $8F00,X       ; STORE THE BYTE - NOT A TERM CMD
93FB 861C           STX $1C           ; PUT X ASIDE
93FD A6FA           LDX $FA           ; GET FLAG
93FF E006           CPX #$06          ; IS IT 6?
9401 F005           BEQ ELIM          ; YES
9403 A61C           LDX $1C           ; GET X BACK
9405 4C9594         JMP CNTAA         ; CONTINUE
9408 A201    ELIM   LDX #$01          ; SUBROUTINE TO ELIMINATE
```

(Continued)

Consider the following program:

```
10 GOTO 50
50 END
```

A simple concatenation would be:

```
10 GOTO 50 : END
```

Any attempt to run the shortened program would result in an Applesoft error. Consider a slight modification of this example:

```
10 GOTO 50
20 J = 5
50 END
```

By appending the program together into one statement, it would appear as:

```
10 GOTO 50 : J = 5 : END
```

Again there would be an Applesoft error. Note that the first part of the statement (GOTO 50) immediately transfers control to a later line number.

Assume for a moment that a called line number table has been constructed, and the 'GOTO 50' statement identified. Recognizing that line 50 could not be eliminated, COMPRESS would preserve it. After compression the program would look like this:

```
10 GOTO 50 : J = 5
50 END
```

While the execution of this short program would not create a computer error, the transfer of control to line 50 takes place *before* the 'J = 5' is executed. So 'J = 5' would *never* be executed at all.

Another condition that the compression program must address is Terminal Commands. Commands that not only transfer control, but do it irrevocably (as shown in the example above), are called Unconditional Terminal Commands. Commands that may or may not transfer control are called Conditional Terminal Commands. Unconditional Terminal Commands include:

GOTO
ON ... GOTO
RETURN
REM (see the discussion later in the article on REM)

Conditional Terminal Commands include:

IF ... GOTO
IF ... THEN GOTO
IF ... THEN ...

Conditional commands behave in a special way. For example:

```
10 IF A = 2 THEN B = 5
20 C = 6
```

Executed without compression, the 'C = 6' assignment will be executed regardless of the outcome of statement 10. If these lines are concatenated, they appear as:

```
10 IF A = 2 THEN B = 5 : C = 6
```

Applesoft will evaluate the 'IF A = 2' part of the line. If it is false, then no further action will take place on that line; control will immediately fall through to the next line. Conditional Terminal Commands are different from Unconditional Terminal Commands because control is transferred immediately *only* if the condition in the statement is false. Nevertheless, COMPRESS cannot append any subsequent statements to terminal commands of any type, because doing so may subvert the programmer's intent.

It is legal, however, for the programmer to concatenate many statements after a conditional command. This is a method of saving space and speeding up execution time, but it can only be done by the programmer. COMPRESS cannot possibly know the initial intent. A Conditional Terminal Command then, is one where control may or may not be passed to the next statement, or to some numbered statement earlier or later in the program.

COMPRESS takes two actions with respect to conditional commands. First, any referenced line number is recorded in a Called Line Number Table so that the line number will not be eliminated. Second, as with any terminal command, there will be nothing appended to the Conditional Terminal Command, unless the programmer added something in the initial program. COMPRESS always appends the last statement to the line. Unconditional Terminal Commands will be treated in exactly the same manner.

REM Statements

Applesoft treats REM in a special way. When a REM token ($B2) is encountered in a program, everything after the REM is considered part of the remark until the $00 delimiter is encountered. The statement

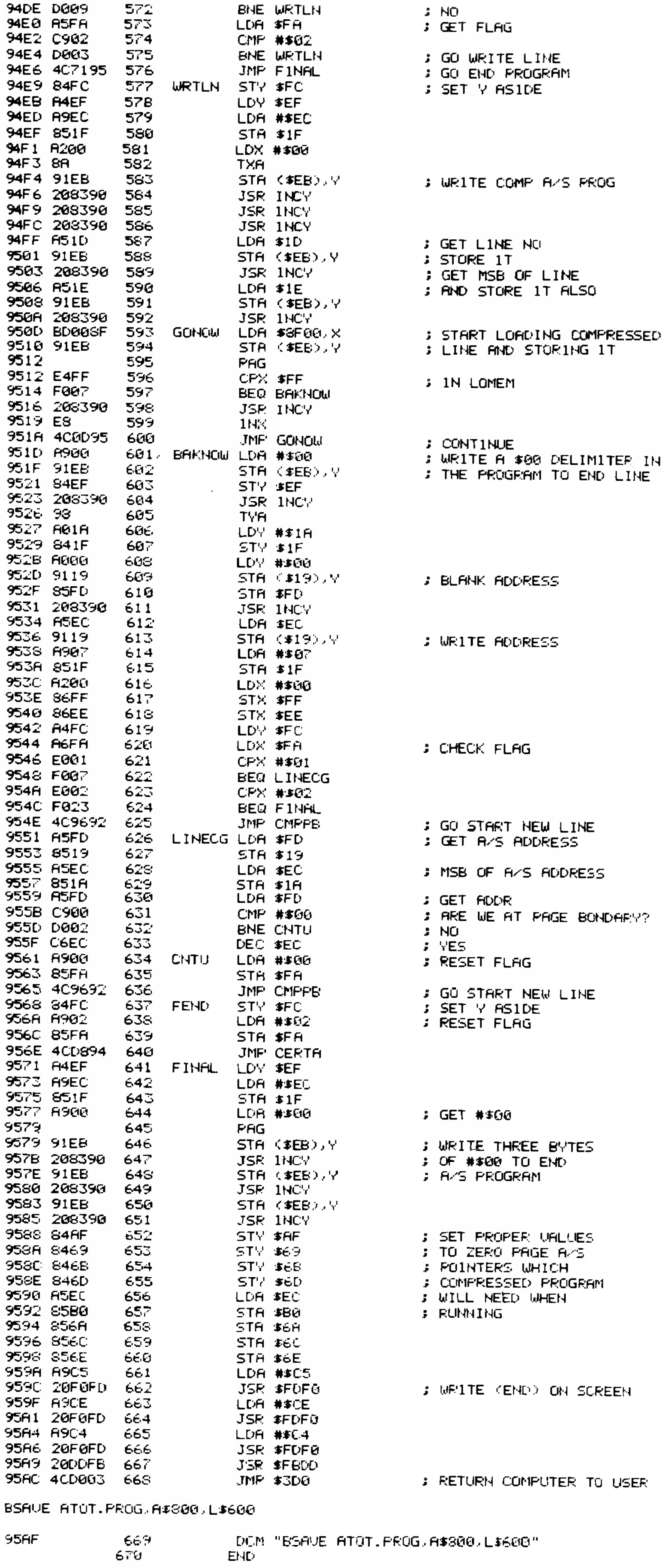
```
10 REM THIS IS A REMARK : J = 10
```

would not be evaluated past the first

Listing 1 *(continued)*

```
940A 86FA           STX $FA        ; EVERYTHING AFTER "REM"
940C A61C           LDX $1C
940E 4C1494         JMP ELIM3
9411 208390  ELIM2  JSR INCY
9414 B106    ELIM3  LDA ($06),Y    ; GET ANOTHER BYTE
9416 C900           CMP #$00       ; IS IT 0?
9418 D0F7           BNE ELIM2      ; NO - LOOP
941A 4C9594         JMP CNTAA      ; YES - REM IS DONE
941D 85ED    GOTO   STA $ED        ; PUT BYTE ASIDE
941F E000           CPX #$00       ; IS X 0?
9421 F00C           BEQ THRU       ; YES
9423 CA             DEX            ; NO - BACK UP
9424 BD008F         LDA $8F00,X    ; GET LAST BYTE
9427 C9C4           CMP #$C4       ; WAS IT 'THEN'?
9429 D003           BNE INCC       ; NO
942B 4C2F94         JMP THRU       ; YES
942E E8      INCC   INX            ; RESET X AGAIN
942F A901    THRU   LDA #$01       ; RESET FLAG
9431 85FA           STA $FA
9433 A5ED           LDA $ED        ; GET ACC BACK
9435 4CF893         JMP PROGWT     ; GO WRITE BYTE
9438 85ED    IF     STA $ED        ; PUT BYTE ASIDE
943A A901           LDA #$01       ; RESET FLAG
943G                PAG
943C 85FA           STA $FA
943E A5ED           LDA $ED        ; GET BYTE BACK
9440 4CF893         JMP PROGWT     ; GO WRITE BYTE
9443 85ED    NXTT   STA $ED        ; PUT BYTE ASIDE
9445 B106    NXTX   LDA ($06),Y    ; GET NEXT A/S BYTE
9447 C93A           CMP #$3A       ; IS IT COLON?
9449 F00A           BEQ LOADM      ; YES - NO VAR NAME
944B C900           CMP #$00       ; IS IT END OF STMT?
944D F006           BEQ LOADM      ; YES - NO VAR NAME
944F 208390         JSR INCY
9452 4C4594         JMP NXTX       ; GO GET ANOTHER
9455 A5ED    LOADM  LDA $ED        ; GET BYTE BACK
9457 4CF893         JMP PROGWT     ; GO WRITE BYTE
945A B106    LET    LDA ($06),Y    ; GET NEXT A/S BYTE
945C 208390         JSR INCY
945F 4CF893         JMP PROGWT     ; GO WRITE IT
9462 E000    REM    CPX #$00       ; IS THIS NEW LINE?
9464 F001           BEQ REMA       ; YES
9466 CA             DEX            ; NO - BACK UP
9467 861C    REMA   STX $1C        ; PUT X ASIDE
9469 A6FA           LDX $FA        ; LOOK AT FLAG
946B E005           CPX #$05
946D D00E           BNE REMC
946F A206           LDX #$06       ; RESET FLAG
9471 86FA           STX $FA
9473 A61C           LDX $1C        ; GET X BACK
9475 E000           CPX #$00       ; IS IT A NEW LINE?
9477 F001           BEQ REMB       ; YES
9479 E8             INX            ; NO - ADVANCE 1
947A 4CF893  REMB   JMP PROGWT     ; GO WRITE BYTE
947D A61C    REMC   LDX $1C        ; GET X BACK
947F B106           LDA ($06),Y    ; GET ANOTHER A/S BYTE
9481 C900           CMP #$00       ; IS IT 0?
9483 F006           BEQ CNTSB      ; YES GO SEE IF END
9485 208390         JSR INCY       ; ADVANCE POINTER
9488 4C7D94         JMP REMC       ; CONTINUE
948B E000    CNTSB  CPX #$00       ; IS THIS A NEW LINE?
948D D005           BNE CNTAX      ; NO
948F 86FA           STX $FA        ; RESET FLAG
9491 4C9692         JMP CMPPB      ; GO GET ANOTHER BYTE
9494 CA      CNTAX  DEX            ; BACK UP X REG
9495 B106    CNTAA  LDA ($06),Y    ; GET ANOTHER A/S BYTE
9497 E0EE           CPX #$EE       ; ARE WE OUT OF ROOM?
9499 F00B           BEQ BACKUP     ; YES - BACKUP
949B E8             INX
949C C900           CMP #$00       ; IS BYTE 0?
949E F016           BEQ ENDEM      ; YES - GO END
94A0 208390         JSR INCY
94A3 4C9E93         JMP PROG       ; CONTINUE
94A6                PAG
94A6 208893  BACKUP JSR DECY       ; BACK UP Y REG
94A9 C418           CPY $18
94AB D0F9           BNE BACKUP     ; CONTINUE
94AD 84FC           STY $FC        ; SET Y ASIDE
94AF A201           LDX #$01       ; RESET FLAG
94B1 86FA           STX $FA
94B3 4CD894         JMP CERTA
94B6 E0EE    ENDEM  CPX #$EE       ; ARE WE OUT OF ROOM?
94B8 F0EC           BEQ BACKUP     ; YES
94BA A93A           LDA #$3A       ; GET COLON
94BC 9D008F         STA $8F00,X    ; WRITE COLON
94BF 8DFF8F         STA $8FFF      ; STORE IN FLAG
94C2 86EE           STX $EE        ; SET X ASIDE
94C4 E8             INX
94C5 86FF           STX $FF
94C7 A6FA           LDX $FA        ; GET FLAG
94C9 E005           CPX #$05
94CB D004           BNE END2       ; TIME TO END?
94CD A200           LDX #$00       ; RESET FLAG
94CF 86FA           STX $FA
94D1 E001    END2   CPX #$01       ; IS FLAG 1?
94D3 F003           BEQ CERTA      ; YES GO WRITE LINE
94D5 4C9692         JMP CMPPB      ; NO CONTINUE
94D8 A6EE    CERTA  LDX $EE        ; GET X BACK
94DA 86FF           STX $FF        ; OVERWRITE POINTER
94DC E000           CPX #$00       ; IS X 0?
```

Listing 1 *(continued)*



```
94DE D009          BNE WRTLN          ; NO
94E0 A5FA          LDA $FA            ; GET FLAG
94E2 C902          CMP #$02
94E4 D003          BNE WRTLN          ; GO WRITE LINE
94E6 4C7195        JMP F1NAL          ; GO END PROGRAM
94E9 84FC   WRTLN  STY $FC            ; SET Y AS1DE
94EB A4EF          LDY $EF
94ED A9EC          LDA #$EC
94EF 851F          STA $1F
94F1 A200          LDX #$00
94F3 8A            TXA
94F4 91EB          STA ($EB),Y        ; WR1TE COMP A/S PROG
94F6 208390        JSR INCY
94F9 208390        JSR 1NCY
94FC 208390        JSR 1NCY
94FF A51D          LDA $1D            ; GET L1NE NO
9501 91EB          STA ($EB),Y        ; STORE 1T
9503 208390        JSR 1NCY           ; GET MSB OF LINE
9506 A51E          LDA $1E            ; AND STORE 1T ALSO
9508 91EB          STA ($EB),Y
950A 208390        JSR 1NCY
950D BD008F GONOW  LDA $8F00,X        ; START LOADING COMPRESSED
9510 91EB          STA ($EB),Y        ; LINE AND STOR1NG 1T
9512               PAG
9512 E4FF          CPX $FF            ; 1N LOMEM
9514 F007          BEQ BAKNOW
9516 208390        JSR INCY
9519 E8            1NX
951A 4C0D95        JMP GONOW          ; CONT1NUE
951D A900   BAKNOW LDA #$00           ; WRITE A $00 DELIM1TER IN
951F 91EB          STA ($EB),Y        ; THE PROGRAM TO END LINE
9521 84EF          STY $EF
9523 208390        JSR 1NCY
9526 98            TYA
9527 A01A          LDY #$1A
9529 841F          STY $1F
952B A000          LDY #$00
952D 9119          STA ($19),Y        ; BLANK ADDRESS
952F 85FD          STA $FD
9531 208390        JSR 1NCY
9534 A5EC          LDA $EC
9536 9119          STA ($19),Y        ; WR1TE ADDRESS
9538 A907          LDA #$07
953A 851F          STA $1F
953C A200          LDX #$00
953E 86FF          STX $FF
9540 86EE          STX $EE
9542 A4FC          LDY $FC
9544 A6FA          LDX $FA            ; CHECK FLAG
9546 E001          CPX #$01
9548 F007          BEQ LINECG
954A E002          CPX #$02
954C F023          BEQ F1NAL
954E 4C9692        JMP CMPPB          ; GO START NEW LINE
9551 A5FD   LINECG LDA $FD            ; GET A/S ADDRESS
9553 8519          STA $19
9555 A5EC          LDA $EC            ; MSB OF A/S ADDRESS
9557 851A          STA $1A
9559 A5FD          LDA $FD            ; GET ADDR
955B C900          CMP #$00           ; ARE WE AT PAGE BONDARY?
955D D002          BNE CNTU           ; NO
955F C6EC          DEC $EC            ; YES
9561 A900   CNTU   LDA #$00           ; RESET FLAG
9563 85FA          STA $FA
9565 4C9692        JMP CMPPB          ; GO START NEW LINE
9568 84FC   FEND   STY $FC            ; SET Y AS1DE
956A A902          LDA #$02           ; RESET FLAG
956C 85FA          STA $FA
956E 4CD894        JMP CERTA
9571 A4EF   FINAL  LDY $EF
9573 A9EC          LDA #$EC
9575 851F          STA $1F
9577 A900          LDA #$00           ; GET #$00
9579               PAG
9579 91EB          STA ($EB),Y        ; WRITE THREE BYTES
957B 208390        JSR 1NCY           ; OF #$00 TO END
957E 91EB          STA ($EB),Y        ; A/S PROGRAM
9580 208390        JSR 1NCY
9583 91EB          STA ($EB),Y
9585 208390        JSR 1NCY
9588 84AF          STY $AF            ; SET PROPER VALUES
958A 8469          STY $69            ; TO ZERO PAGE A/S
958C 846B          STY $6B            ; PO1NTERS WHICH
958E 846D          STY $6D            ; COMPRESSED PROGRAM
9590 A5EC          LDA $EC            ; WILL NEED WHEN
9592 85B0          STA $B0            ; RUNNING
9594 856A          STA $6A
9596 856C          STA $6C
9598 856E          STA $6E
959A A9C5          LDA #$C5
959C 20F0FD        JSR $FDF0          ; WR1TE (END) ON SCREEN
959F A9CE          LDA #$CE
95A1 20F0FD        JSR $FDF0
95A4 A9C4          LDA #$C4
95A6 20F0FD        JSR $FDF0
95A9 20DDFB        JSR $FBDD
95AC 4CD003        JMP $3D0           ; RETURN COMPUTER TO USER

BSAVE ATOT.PROG,A$800,L$600

95AF               DCM "BSAVE ATOT.PROG,A$800,L$600"
                   END
```

REM. Everything, including the colon and the second statement, is considered part of the remark. Nothing can therefore be appended to REM statements; they are considered terminal commands.

REM statements will be entirely eliminated by COMPRESS to save space unless they are on called line numbers. Programmers who want maximum compression should avoid branching to REM statements. COMPRESS cannot eliminate a REM statement if it is the result of a branch from another line; however, it will eliminate everything after the REM ($B2), reducing the statement to just one token.

Executing COMPRESS

Three steps are required to execute COMPRESS:

1. Type BRUN COMPRESS
2. When the cursor reappears, LOAD your Applesoft program.
3. When the cursor appears again, type CALL 36875.

COMPRESS will respond with:

```
END OF PASS #1
END
```

Your program is now compressed! To see how much memory-saving compression took place, try it again, but follow these steps:

Type BRUN COMPRESS

When the cursor reappears, LOAD your Applesoft program.

Type CALL –151

Type AF.B0

The computer will give you the address of the highest memory location being used by your current program. The high order byte is last, and the low order byte first. If, for example, the computer responds with:

```
00AF - AA
00B0 - 1B
```

then the top of your Applesoft program resides at location $1BAA, and occupies the space between $0800 (where all Applesoft programs start) and $1BAA.

Now, type CTRL-C to get back into Applesoft. Again type CALL 36875 to invoke COMPRESS. After COMPRESS is finished, enter the monitor again with a CALL –151, type AF.B0 again, and record the new memory extents.

By converting these addresses to decimal you can see how effective COMPRESS is! (More about that later.)

One final note on running COMPRESS. Once you have typed BRUN COMPRESS and the cursor reappears, it is unnecessary to execute this step again as long as the computer is not turned off, or you do not run any other machine-language programs that operate up near HIMEM. Simply type CALL 36875 to invoke COMPRESS. But don't forget that COMPRESS occupies nine pages of RAM memory, which may be valuable space depending on the function you wish to perform. A PR#6 will erase COMPRESS.

The Three Steps of COMPRESS

Step 1: Set HIMEM

Program steps 3 to 7 tell the computer that COMPRESS is in the computer, located just below DOS between locations $9000 and $95FF. Also, the space $8D00 and $8EFF is reserved for the Called Line Number Table, and $8F00 to $8FFF for the Compressed Statement Buffer. HIMEM is an Apple II pointer whose address is stored at location $73 and $74 on the zero-page of memory. These two bytes will be set to $00 and $8D, respectively, following the format we have been using all along of placing the least significant byte first.

After HIMEM is set, control is returned to the user and the Applesoft cursor (]) reappears. The 'JMP $3D0' accomplishes this. At that location, Applesoft is reinvoked.

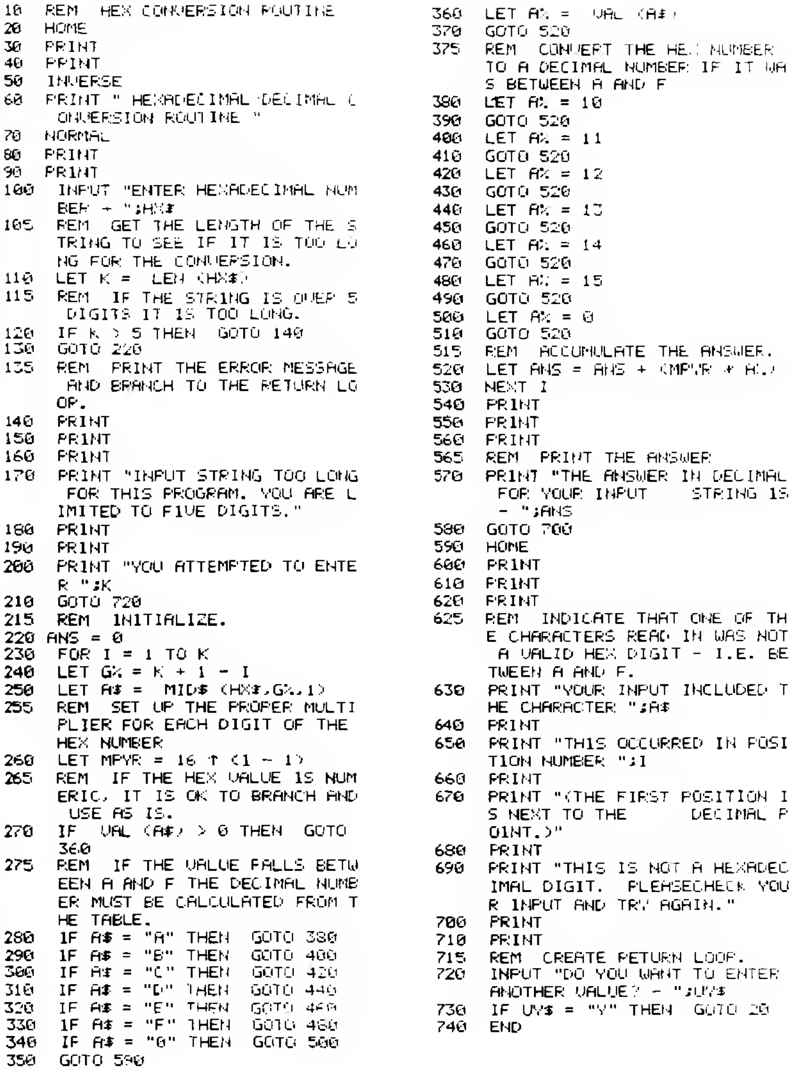
Step 2: TABLECREATE

Lines 8 to 62 initialize the memory pointers for COMPRESS. While this may not be necessary immediately, all of the initialization routines are contained here.

Lines 63 to 70 are subroutine INCY, which is used throughout the program to increment the Y register by 1. If the value of the Y register is forced to $00 (that is, it passes $FF), then the high-order byte of the sixteen bit address being incremented (which is stored in $1F and loaded into the X register) is itself incremented with the command INC $00,X.

Lines 71 to 87 are an exit from the program and contain the ERROR subroutine. Notice that line 87 returns control to the user with a JMP to $3D0. Prior to that, however, the message ERROR #xx appears on the screen. COMPRESS recognizes three error messages:

***Listing 2:* HEXDEC**



```
10   REM  HEX CONVERSION ROUTINE
20   HOME
30   PRINT
40   PRINT
50   INVERSE
60   PRINT " HEXADECIMAL DECIMAL C
      ONVERSION ROUTINE "
70   NORMAL
80   PRINT
90   PRINT
100   INPUT "ENTER HEXADECIMAL NUM
     BER - ";HX$
105  REM  GET THE LENGTH OF THE S
     TRING TO SEE IF IT IS TOO LO
     NG FOR THE CONVERSION.
110  LET K =  LEN (HX$)
115  REM  IF THE STRING IS OVER 5
      DIGITS IT IS TOO LONG.
120  IF K > 5 THEN  GOTO 140
130  GOTO 220
135  REM  PRINT THE ERROR MESSAGE
      AND BRANCH TO THE RETURN LO
     OP.
140  PRINT
150  PRINT
160  PRINT
170  PRINT "INPUT STRING TOO LONG
      FOR THIS PROGRAM. YOU ARE L
     IMITED TO FIVE DIGITS."
180  PRINT
190  PRINT
200  PRINT "YOU ATTEMPTED TO ENTE
     R ";K
210  GOTO 720
215  REM  INITIALIZE.
220 ANS = 0
230  FOR I = 1 TO K
240  LET G% = K + 1 - I
250  LET A$ =  MID$ (HX$,G%,1)
255  REM  SET UP THE PROPER MULTI
     PLIER FOR EACH DIGIT OF THE
     HEX NUMBER
260  LET MPYR = 16 ^ (I - 1)
265  REM  IF THE HEX VALUE IS NUM
     ERIC, IT IS OK TO BRANCH AND
      USE AS IS.
270  IF  VAL (A$) > 0 THEN  GOTO
     360
275  REM  IF THE VALUE FALLS BETW
     EEN A AND F THE DECIMAL NUMB
     ER MUST BE CALCULATED FROM T
     HE TABLE.
280  IF A$ = "A" THEN  GOTO 380
290  IF A$ = "B" THEN  GOTO 400
300  IF A$ = "C" THEN  GOTO 420
310  IF A$ = "D" THEN  GOTO 440
320  IF A$ = "E" THEN  GOTO 460
330  IF A$ = "F" THEN  GOTO 480
340  IF A$ = "0" THEN  GOTO 500
350  GOTO 590
360  LET A% =  VAL (A$)
370  GOTO 520
375  REM  CONVERT THE HEX NUMBER
     TO A DECIMAL NUMBER IF IT WA
     S BETWEEN A AND F
380  LET A% = 10
390  GOTO 520
400  LET A% = 11
410  GOTO 520
420  LET A% = 12
430  GOTO 520
440  LET A% = 13
450  GOTO 520
460  LET A% = 14
470  GOTO 520
480  LET A% = 15
490  GOTO 520
500  LET A% = 0
510  GOTO 520
515  REM  ACCUMULATE THE ANSWER.
520  LET ANS = ANS + (MPYR * A%)
530  NEXT I
540  PRINT
550  PRINT
560  PRINT
565  REM  PRINT THE ANSWER
570  PRINT "THE ANSWER IN DECIMAL
      FOR YOUR INPUT    STRING IS
      - ";ANS
580  GOTO 700
590  HOME
600  PRINT
610  PRINT
620  PRINT
625  REM  INDICATE THAT ONE OF TH
     E CHARACTERS READ IN WAS NOT
      A VALID HEX DIGIT - I.E. BE
     TWEEN A AND F.
630  PRINT "YOUR INPUT INCLUDED T
     HE CHARACTER ";A$
640  PRINT
650  PRINT "THIS OCCURRED IN POSI
     TION NUMBER ";I
660  PRINT
670  PRINT "(THE FIRST POSITION I
     S NEXT TO THE      DECIMAL P
     OINT.)"
680  PRINT
690  PRINT "THIS IS NOT A HEXADEC
     IMAL DIGIT.  PLEASECHECK YOU
     R INPUT AND TRY AGAIN."
700  PRINT
710  PRINT
715  REM  CREATE RETURN LOOP.
720  INPUT "DO YOU WANT TO ENTER
     ANOTHER VALUE? - ";UV$
730  IF UV$ = "Y" THEN  GOTO 20
740  END
```

ERROR #01 or ERROR #02: A line number that exceeds $FFFF has been detected. This can occur during the conversion routine if a partial line number is followed by an illegal code in a branching statement, or a line number is specified that exceeds decimal 65535. Note: line numbers greater than 63999 will generate an Applesoft error. Tokens $30 through $39 are the only legal numbers that can be used with a program line to represent a line number. The program (uncompressed) should be RUN to locate the source of the Applesoft error.

ERROR #03: More than 255 Called Line Numbers appear in the program. Note that these do not have to be different numbers to generate an error. More than 255 repetitions of the same callec line number will create an error condi tion. COMPRESS is limited to th number of 16-bit (two-byte) addresse that it can fit between $8D00 an $8EFF, minus 1.

Line 88, which appears severa times throughout the program, loads single byte of the uncompressed Apple soft program. The command 'LD ($06),Y' instructs the computer to loa into the accumulator the contents of memory location whose 16-bit (twc byte) address is stored starting at loc tion $06, least significant byte firs The address at $06 and $07 (initiall $00 $08, or $0800) is offset or indexe by the contents of the Y register, intiall $0, and incremented by subroutin INCY described above. When Y = $A for example, the address read into th accumulator will be $08A3.

Lines 90 to 141 evaluate the byte read-in, and if indeed it is a called line number, begin the conversion routine.

Applesoft stores everything as tokens except for addresses and actual line numbers, which are stored in hex. Even called line numbers appear as tokens, occupying the range $30 to $39. Applesoft interprets tokens $30 to $39 as the decimals 0 through 9 with one byte (token) per digit. The statement 'GOTO 456' would appear as '$AB $34 $35 $36'. Since the actual line numbers in the program appear as hexadecimal numbers and the called line numbers do not, COMPRESS must go through a conversion routine. The steps required are:

a. Subtract $30 from the token.

b. If the value of the result is less than zero, then the initial token was less than $30, and the result could not have been a called line number.

c. If the value is higher than $0A (hex for 10), then the token was greater than $39, and was not part of the line number.

d. The field is thus narrowed down to the tokens that were originally in the range of $30 to $39, and that will, after subtracting $30, create a decimal digit between 0 and 9. The actual conversion of these decimal digits to hex can then begin.

Lines 142 through 220 convert these decimalized tokens to their hexadecimal equivalents. Each digit in a decimal number has a corresponding hex value. The units digit, for example, has a value of $1 for each decimal 1. Decimal 9 therefore equals $9. The tens digit has a hex value of $A for each decimal 1. Decimal 40, therefore, has the same value as $A + $A + $A + $A. The hundreds digit has the value of $64 for each decimal 1: the thousands $3E8, and the ten-thousands $2710.

Since the computer can add hexadecimal numbers quite easily, a register is incremented by the hex equivalent of each decimal digit as many times as there is value in the decimal number. For example, the line number 30000 would be converted by incrementing the ten thousands equivalent three times: $2710 + $2710 + $2710.

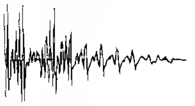
Lines 221 through 251 write the called line number to the storage area now that it has been converted to hex. Since the line numbers in the program that precede each statement are already hex, the numbers can now be directly matched. This takes place after the entire program is scanned.

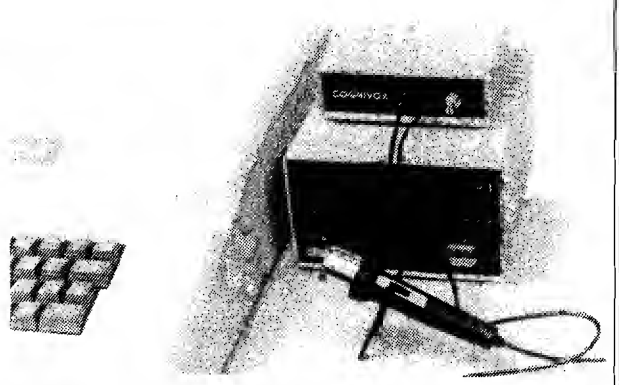
Lines 252 through 280 print the message END OF PART #1.

Step 3: APPEND

Lines 281 through 302 initialize pointers and registers for the Append process. Certain values carried over from step 2, Tablecreate, are not changed, but many working registers must be reinitialized for Append.

Lines 303 to 330 read and evaluate each byte of the Applesoft program and take certain action if $00 appears. One occurrence of $00 indicates a line delimiter. If two appear, the computer must test to see what the next (third) byte may be. There may be two $00 bytes because the low-order byte of the address of the next statement in memory happens to be $00. It is also





possible that the computer has encountered the end of the program. This condition is detected in Applesoft by the occurrence of three $00 bytes.

Lines 331 through 404 search the Called Line Number Table residing at $8D00 to $8EFF and preserve the line number if it appears. All elements must be checked since the called line numbers are stored in the order they occur in the program.

Checking a 16-bit address with an 8-bit processor like the 6502 involves some complexity. The low-order bytes of the Called Line Number Table are checked against the low-order byte of the line number being read. If there is a match, which is quite likely, the high-order bytes are checked. If they match also, the line number must be saved.

Lines 405 through 547 look for terminal commands, REM statements, etc., and act to terminate a line if any are found.

Lines 548 through 554 back up if more than 239 bytes have been written and a natural end-of-line has not been located. In such cases the computer must search backwards until the previous end of statement is encountered.

Lines 579 to 638 write the compressed statement to the normal RAM storage location for Applesoft programs ($800 upward). Note that after compression the program is written directly over the original uncompressed program. Since COMPRESS can only shorten a program and can never lengthen it, there is never danger that a program will be written over part of the unread original program.

Lines 639 through 669 terminate the COMPRESS program and write the final pointers into the normal Applesoft zero-page locations, as well as print 'END' on the screen.

Line 670 returns control to the user with a JMP to $3D0.

Final Notes

COMPRESS works with most Applesoft programs. Certain conventions used by some programmers may make COMPRESS unable to do its job. Some authors have sprinkled $20 tokens ($20 is the Applesoft representation for space) into called line number areas of their programs to discourage unauthorized copying. While this is not the only condition that will 'fool' COMPRESS, it is typical of unconventional programming techniques that COMPRESS cannot handle.

Included with this article as listing 2 is an Applesoft program named HEXDEC. This program can serve two purposes: it is a good test program to run through COMPRESS, and it will also enable the user to convert the addresses located at $AF and $B0 to decimal. Then the user will be able to see how much memory usage has been eliminated through the use of COMPRESS.

To use HEXDEC, enter the values located at $AF.B0 (reverse them first of course) and get the decimal equivalent. Subtract the equivalent of $800 (decimal 2048) to find out how many bytes of storage your program takes. Do the same after a compression run and then calculate the difference.

HEXDEC itself occupies 1906 bytes of storage before compression, and 964 after ($0F72 - $800 = decimal 1906; $0BC4 - $800 = decimal 964). This amounts to a 49.5% memory reduction!

While admittedly this is an extreme example, which was exaggerated solely to show what COMPRESS can do, the basic idea is not far-fetched. The programmer now has the liberty to write many REM statements and put one statement per line for later editing if necessary.

A wordy original Applesoft program now does not exact an execution penalty. COMPRESS can optimize the code prior to execution. All debugging should of course be done before compression. Leaving the original program intact allows the user to make later changes as necessary, and then recompress the code.

One final note of caution: the name under which the COMPRESSed program is saved to disk must be different than the original program or the original will be lost. One suggestion in this regard is to add ".COMP" to the old program name when SAVEing the compressed version. For example PAYROLL could be the original lengthy version, and PAYROLL.COMP the compressed code.

The compilers' promise of dramatically increased execution speed has not been fulfilled to the extent that users expected. And the compiled code may actually require geater memory space. With COMPRESS, you can achieve many of the same goals at substantially lower cost, while preserving all of the important features of the Applesoft language originally envisioned by the designers of the Apple II.

Barton M. Bauers may be contacted at 30 Hillock Drive, Wallingford, Connecticut 06492.

MICRO

MICRObits

(Continued from page 38)

Strictly Commodore

The bi-monthly software-oriented publication entirely devoted to VIC-20, PET, CBM, and SuperPET owners. Each issue contains four to six full-length entertaining programs, detailed product reviews, programming tips, plus latest Commodore news. Subscription $15; sample $2.

Strictly Commodore
47 Coachwood Place N.W.
Calgary, Alta.
Canada T3H 1E1

PET Joystick Interface

Connects directly to all PET/CBM computers. Allows PET to accept either Apple joysticks/paddles or Atari joysticks. No assembly required. Ready to plug into the user port. Software provided. Immediate delivery. Only $49.95. Michigan residents add 4% sales tax.

Systems Corp.
1 Edmund Place
Ann Arbor, MI 48103

OSI Super Defender

Super Defender is an all-machine-code game just like the arcades. The mountains roll by as your scanner shows what's coming at you. Protect your humanoids from being snatched by the landing crafts. $14.95 for tape or 5¼" disk.

DMP Systems
319 Hampton Blvd.
Rochester, NY 14612

Books You'll Need

6509 Software Design - $12.95, *Programming and Interfacing the 6502* - $16.95, *6809 Microcomputer Programming & Interfacing* - $14.95, *68000: Principles & Programming* - $14.95, *Advanced 6502 Interfacing* - $12.95. Add $2.75 handling. Foreign add $7.00. Texas residents add 5% tax. Send for free list.

Phil's Books
Dept. MC1
P.O. Box 120964
Arlington, TX 76012

MICRO

It's like having an Uncle in the Business

All Prices Discounted. Get an additional 3%*

ADVENTURE INTERNATIONAL	
Commbat	$22.00
Tunnel Terror	$25.00
War	$22.00
Rear Guard	$25.00

NEW GRAPHIC ADVENTURES	
S.A.G.A. 1 Adventureland	$25.00
S.A.G.A. 2 Pirates Adv.	$25.00
S.A.G.A. 3 Mission Imp.	$25.00
S.A.G.A. 4 Voodoo Castle	$25.00

AUTOMATED SIMULATIONS	
Fore!	$25.00
Crush, Crumble & Chomp	$25.00
Tues. Morn Quarterback	$25.00

AVALON HILL	
Computer Aquire	$22.00
Computer Stocks & Bonds	$22.00

BEZ	
Bez Off	$27.00

BRODERBUND	
Choplifter	$29.00
Serpentine	$29.00
Deadly Secrets Hi-Res	$29.00

CPU SOFTWARE	
Oil Rig	$33.00
Human Fly	$33.00

COMPUTEK	
Solitaire	$25.00

CAVALIER	
Bug Attack	$25.00
Microwave	$29.00

DATAMOST	
Gin Rummy	$29.00
Pig Pen	$25.00
Kids & The Apple (Book) For Kids From 6 to 60 +	$19.00
Casino	$33.00
Swashbuckler	$29.00
Solo Poker	$25.00
Baseball	$25.00
Missing Ring	$25.00
Country Fair	$25.00
Thief	$25.00
Snack Attack	$25.00

DATASOFT	
Tumble Bugs	$25.00

GEBELLI	
Neptune	$25.00
Lazer Silk	$25.00
Zenith	$29.00
Russki Duck	$25.00

INFOCOM	
Zork I	$33.00
Zork II	$33.00
Deadline	$42.00

MICRO LAB	
Roach Hotel	$28.00
Peeping Tom	$28.00

MUSE	
Firebug	$22.00
Frazzle	$22.00

ON LINE SYSTEMS	
Hi Res Asteroid	$22.00
Hi Res Mystery House	$22.00
Roadwork	$33.00
Jawbreaker (Improved)	$25.00
Mouskattack	$28.00
Ulysses/Golden Fleece	$28.00
Wiz & Princess	$28.00
Frogger	$28.00
Cannonball Blitz	$28.00

PICCADILLY	
Star Blaster	$25.00
Ribbit	$25.00

ROCKROY	
Conglomerates Collide	$33.00

SIRIUS SOFTWARE	
Twerps	$25.00
Dark Forest	$25.00
Jellyfish	$25.00
Minotaur	$29.00
Bandits	$29.00

SIR-TECH	
Wizardry	$39.00
Knights of Diamonds (Scenario #2 for Wizardry)	$29.00

STRATEGIC SIMULATIONS	
Cytron Masters	$33.00
Cartels & Cutthroats	$33.00
Shattered Alliance	$49.00
Computer Bismark	$49.00
Computer Ambush 2nd Ed.	$49.00

SUB-LOGIC	
Flight Simulator	$28.00
Pin-Ball/A Great Pinball Game	$25.00
Saturn Navigator	$29.00

SYNERGISTIC SOFTWARE	
Nightmare Gallery	$29.00
U-Boat Command	$29.00
Atlantis	$33.00
Escape From Arcturus	$29.00

TURNKEY SOFTWARE	
Ceiling Zero	$25.00
The Gauntlet	$29.00

ACCOUNTING	
Accounting Plus	$318.00
Bus. Bookkeeping Sys.	$314
Mod #1 Gen. Ledger	$199.00
Mod #2 Acct. Rec.	$199.00
Financial Partner	$189.00
Gen Ledg & Pay	$389.00
Invoice Factory	$166.00
Prof Time & Billing	$328.00

COMMUNICATIONS	
ASCII Express	$66.00
Data Capture 4.0	$54.00
Data Capture 4.0 80 Col.	$75.00

COMPILERS	
Expediter II +	$113.00
Tasc Compiler	$139.00

DATA BASE	
D.B. Master	$178.00
Filefax	$159.00
The General Manager	$124.00
PFS: Personal Filing Syst	$99.00
Versa Form	$309.00

DATA BASE APPLICATION DEV.	
D. Base II	$524.00

DATA BASE UTILITY	
D.B. Master Stat Pak.	$83.00
D.B. Master Utility Pak I	$83.00
D.B. Master Utility Pak II	$83.00

DATA ENTRY	
Datastar	$235.00

EDUCATIONAL	
Counting Bee	$25.00
Decimals	$39.00
The Learning System	$124.00
Master Type	$33.00
SAT Word Attack	$39.00
Speed Reading	$149.00
Whole Brain Spelling	$30.00

FINANCIAL MODEL/PLAN.	
Asset Manager	$159.00
Calcstar	$159.00
Creative Financing	$159.00
Visicalc	$191.00
Wall Streeter	$229.00

GRAPHICS	
Arcade Machine	$37.00
Budges 3D Graphics	$33.00
E-Z Draw 3.3	$39.00
Graphics Magician	$49.00
Hi-Res Secrets	$99.00

HOUSEHOLD ACCOUNTING	
The Home Accountant	$65.00

MATHEMATICS/ENGINEERING	
Mathmagic	$74.00

MAILING LIST	
First Class Mail	$59.00
The Mailroom	$29.00

MUSIC/SOUND/SPEECH	
Apple Talker	$25.00
The Voice	$33.00

PROP. MGMT/REAL ESTATE	
R.E.A.P (The Best I Have Seen)	$103.00

TAX	
Loophole Louie Loves The Tax Beater	$105.00

UTILITY	
Back It Up	$44.00
Mem. Mgmt. Syst. II	$39.00
Multi Disk Catalog	$20.00
Super Disk Copy III	$25.00

UTILITY—PROGRAMMING	
Apple Doc	$39.00
Apple Mechanic	$25.00
Apple Spice	$25.00
Bag of Tricks	$33.00
Menu Generator	$33.00
Programming Aids	$74.00
The Bug	$39.00

UTILITY—PRINTER	
Zoom Graphics	$33.00

WORD PROCESSOR	
Exec. Secretary	$189.00
Format II	$259.00
Screenwriter II	$99.00
Wordstar Req Softcard	$249.00
Write-On! II Loaded with Features Found in W/P S/W costing more than double the price	$99.00

FREE GIFT

with any software order over $40.00

With any order of $100 or more you may select any one of the following at these fantastic prices!

Computer Aquire	$18.00
Bez Off	$24.00
Choplifter	$25.00
Serpentine	$25.00
Deadly Secrets Hi-Res	$25.00
Pig Pen	$19.00
For Kids & The Apple (Book)	$15.00
Swashbuckler	$21.00
Thief	$19.00
Snack Attack	$19.00
Zork I	$29.00
Zork II	$29.00
Nightmare Gallery	$25.00
Sat Word Attack	$33.00
Speed Reading	$129.00
Hi-Res Secrets	$85.00
R.E.A.P. (The Best I Have Seen)	$85.00
Tax Beater	$85.00
Back It Up	$39.00
Bag of Tricks	$28.00
Menu Generator	$28.00

With any order of $200 or more you may select any one of the following at these Super Low Prices!

Bus Bookkeeping Sys	$275.00
Versa Form	$259.00
D Base II	$450.00
Datastar	$180.00
Visicalc	$175.00
Wall Streeter	$200.00
Format II	$229.00

YANKEE PEDDLER
10810 Cantara Street
Sun Valley, CA 91352
(213) 896-3642

*Take an additional 3% discount if a check (certified, cashiers or personal) accompanies the order. Please allow 14 days for personal check clearance. We also accept MasterCard and Visa. Please enclose both number and exp. date. Please add 5% for shipping & handling. Minimum shipping & handling charge is $1.50. Foreign order minimum $75.00 plus 12% shipping & handling (except Canada). If we deem that shipping & handling charges are excessive, we will return the excess to you. Calif residents add 6.5% tax.

PRICES SUBJECT TO CHANGE WITHOUT NOTICE.

Book Reviews

by John Valente

68000 Assembly Language Programming, by Gerry Kane, Doug Hawkins and Lance Leventhal. Osborne/McGraw-Hill, 1981. ISBN: 0-931988-62-4. Paper, 680 pages, $16.99.

68000 Microprocessor Handbook, by Gerry Kane. Osborne/McGraw-Hill, 1981. ISBN: 0-931988-41-1. Paper, 200 pages, $6.99.

These two volumes are the only widely available texts on the 68000. Fortunately, they are excellent. As of this writing, the only 68000 computers available are expensive business-oriented systems. (*Editor's note:* Motorola has announced a 68000 Educational Computer Board ($495.00) that will make it much easier to begin learning the 68000.) These two books should thoroughly prepare you for using the full power of the 68000. Even a casual glance shows this processor to be truly a "new generation" device.

Both volumes continue the tradition of quality in the Osborne series. They are well-written, concise, and clearly printed. Major points are in boldface, so the reader may find desired information quickly. Diagrams abound. Extensive indexes are included. The authors stick to the point and do not waste words; the reader should have some familiarity with microprocessor architecture and machine language.

68000 Assembly Language Programming begins with a discussion of the merits and deficiencies of assembly language. Some readers may have read similar material in other books on assembly-level programming. Still, the material is worth reading as an overview. A detailed description of a typical assembler follows; it is a very good tutorial, and applies to any processor. It also acquaints the reader with the terminology used throughout the text.

Chapter 3 gets to the heart of the matter: 68000 Machine Structure and Assembly Language. The author describes clearly the user/supervisor operating modes and the register set. Diagrams help to explain the various addressing modes, showing exactly what is happening in the machine.

The next few chapters develop familiarity with the instruction set through example programs that get progressively more difficult. The problems are typical of those selected in assembly-language texts: simple arithmetic, code conversion, tables, and lists. The reader, who may have seen these problems before, sees how they can be solved in the 68000 environment.

The third section presents "Advanced Topics" such as parameter passing and subroutines. The 68000 is particularly adept at these applications. Powerful instructions facilitate relocatable and re-entrant routines. I would have liked more coverage of the LINK and UNLINK instructions, which are both new and apparently very useful.

The next three chapters cover I/O using the 6821 PIA and the 6850 ACIA. Projects range from controlling a single LED to an A/D converter. The use of these chips is not new, and some readers may not need to read this section. However, the material is relevant, since new I/O chips for the 68000 are not yet available. In fact, the 68000 has special signals to communicate with the "older generation" 6800-series devices.

Chapter 15 is an important one, dealing with exception processing. Most of us have used NMI and IRQ interrupts, but these are merely two members of a whole class of exceptions. Others are bus and address error, reset, illegal operations, trace, trap, and multiple vectored interrupts. There is even a provision for extending the instruction set. Exception processing can service many peripherals in a very sophisticated system. The material covered here is complex, but close study will make it possible to take full advantage of the 68000.

Section 4 covers software development, from problem definition through testing and debugging. The insights presented here are essential to good programming practice, particularly for those inclined to jump from vague ideas to coding. The summary of typical errors is particularly enlightening, and can save you hours of frustration. In a rare departure from their serious tone, the authors offer some good advice: "Sometimes the following approach may be your best bet: turn off the computer, have a beer, and let your brain rest." (p. 19-21)

The remainder of the book alone would be worth the price. Each instruction is thoroughly described in alphabetical order. Accompanying diagrams show precisely the flow of data between registers and memory. Charts indicate the op-codes for the various addressing modes. This is very handy for quick reference while learning a new instruction set. And if that's not enough, there are three tabular appendices: the first is a listing of the instruction set by mnemonic that includes number of bytes, clock cycles, effects on registers, and a functional summary. The second is an alphabetical listing by mnemonic, with op-codes, byte count, and clock cycles. The third lists the op-code in ascending numerical order. These listings are a gold mine of information for the serious programmer.

I did notice a few errors in several programs and schematic diagrams, but they are minor. Each chapter concludes with a set of problems and a list of pertinent references. Some readers might wish the authors had omitted the generalized chapters on software development for the sake of a lower price. I think the price would be justified even without such material, given the quality of the text. Because the 68000 is a complex device, this book is not for casual reading. Fortunately, the authors and editors have made the work as painless as possible.

The *68000 Microprocessor Handbook* is for the hardware designer and advanced programmer. If a microprocessor data sheet intimidates you, I do not recommend this book. On the other hand, if you can work from a very compact reference, and particularly if you are developing system hardware, this one is for you. Every pin and signal of the 68000 is concisely described. Diagrams are used extensively for bus cycle timings and data movement between registers and memory. There is no tutorial information on the instruction set, and no program examples are given. However, there is a brief description of the instruction groups, with emphasis on the more unusual ones. Two concise appendices list the entire instruction set by mnemonic and by opcode, as in the other book.

I especially liked the two-page introduction, which gives the reader a feel for the uniqueness of the 68000 and compares it to the 8086 and the Z8000. Since many of the new machines will use 16-bit processors, this will help you evaluate the various CPUs. The introduction convinced me that the 68000 is potentially the most powerful of the three.

The conventions used in the diagrams are clearly described before the text begins. A functional overview of the 68000 precedes chapters on bus timing and operation, exception processing, addressing modes and interfacing to 6800 peripherals. Selected data sheets are included at the end.

The text is very detailed, and very technical. The emphasis is on hardware design; all the material you need for programming can be found in the other volume. Only if you are capable of programming from a terse listing of a new instruction set should you rely on this volume for software development. Nonetheless, it is an excellent "standalone" guide to the 68000. For the experienced designer, it lives up to its title as a Handbook.

When you read about the 68000, I think you will be impressed. Users of the 6502 and 6809 will see some basic similarities, such as memory-mapped I/O and extensive addressing modes. However, the 68000 is a more complex processor with many new features. Its speed, 32-bit registers, and provision for multiple users, put it into the minicomputer class. Many analysts see a lack of software as a stumbling block to its acceptance. Perhaps readers of these two books will help solve this problem.

MICRO

HDE inc.

BOX 120
ALLAMUCHY, N.J. 07820
201-362-6574

HUDSON DIGITAL ELECTRONICS INC.

THE TASK* MASTERS

HDE supports the *TIM, AIM, SYM and KIM (TASK) with a growing line of computer programs and peripheral components. All HDE component boards are state-of-the-art 4½" x 6½", with on board regulation of all required voltages, fully compatible with the KIM-4 bus.

OMNIDISK 65/8 and 65/5
Single and dual drive 8" and 5¼" disk systems. Complete, ready to plug in, bootstrap and run. Include HDE's proprietary operating system, FODS (File Oriented Disk System).

DM816-M8A
An 8K static RAM board tested for a minimum of 100 hours and warranted for a full 6 months.

DM816-UB1
A prototyping card with on-board 5V regulator and address selection. You add the application.

DM816-P8
A 4/8K EPROM card for 2708 or 2716 circuits. On board regulation of all required voltages. Supplied without EPROMS.

DM816-CC15
A 15 position motherboard mounted in a 19" RETMA standard card cage, with power supply. KIM, AIM and SYM versions.

DISK PROGRAM LIBRARY
Offers exchange of user contributed routines and programs for HDE Disk Systems. Contact Progressive Computer Software, Inc. for details.

HDE DISK BASIC
A full range disk BASIC for KIM based systems. Includes PRINT USING, IF . . . THEN . . . ELSE. Sequential and random file access and much more. $175.00

HDE ADVANCED INTERACTIVE DISASSEMBLER (AID)
Two pass disassembler assigns labels and constructs source files for any object program. Saves multiple files to disk. TIM, AIM, SYM, KIM versions. $95.00

HDE ASSEMBLER
Advanced, two pass assembler with standard mnemonics. KIM, TIM, SYM and KIM cassette versions. $75.00 ($80.00 cassette)

HDE TEXT OUTPUT PROCESSING SYSTEM (TOPS)
A comprehensive text processor with over 30 commands to format and output letters, documents, manuscripts. KIM, TIM and KIM cassette versions. $135.00 ($142.50 cassette)

HDE DYNAMIC DEBUGGING TOOL (DDT)
Built in assembler/disassembler with program controlled single step and dynamic breakpoint entry/deletion. TIM, AIM, SYM, KIM AND KIM cassette versions. $65.00 ($68.50 cassette)

HDE COMPREHENSIVE MEMORY TEST (CMT)
Eight separate diagnostic routines for both static and dynamic memory. TIM, AIM, SYM, KIM and KIM cassette versions. $65.00 ($68.50 cassette)

AVAILABLE DIRECT OR FROM THESE FINE DEALERS:

Progressive Computer Software
405 Corbin Road
York, PA 17403
(717) 845-4954

Johnson computers
Box 523
Medina, Ohio 44256
(216) 725-4560

Falk-Baker Associates
382 Franklin Avenue
Nutley, NJ 07110
(201) 661-2430

Perry Peripherals
P.O. Box 924
Miller Place, NY 11764
(516) 744-6462

Lux Associates
20 Sunland Drive
Chico, CA 95926
(916) 343-5033

Laboratory Microcomputer Consultants
P.O. Box 84
East Amherst, NY 14051
(716) 689-7344

Reviews in Brief

Product Name:	**SIGMON**
Equip. req'd:	TRS-80C Color Computer with 16K memory
Price:	$29.95
Manufacturer:	Datasoft, Inc. 19519 Business Center Dr. Northridge, California 91324

Description: *SIGMON*, a comprehensive assembly-language monitor for the Color Computer, is distributed on cassette. This versatile monitor incorporates a full disassembler and a line-by-line assembler, which doubles as a memory modify mode; it allows decimal, hexadecimal, and literal text to be entered while inserting the correct binary values in memory. Other abilities include the following: display memory in an 8-column format, showing both hexadecimal and ASCII characters. An optional display shows graphics characters for that range of binary numbers. The LIST command displays memory contents by memory location in a 25-column format. A data-search routine searches for text, hexadecimal values, or hexadecimal values specified as assembly-language mnemonic string.

Pluses: The large number of functions makes a very capable package. Extensive documentation is furnished, including a full assembly dump with comments. In particular, the string search is powerful and versatile.

Minuses: Some points of syntax are not covered in the instructions. The approximately 6.5K-byte program is not available in ROM, and is an inconvenient size for user-loading in ROM.

Skill level required: The user must be moderately familiar with assembly language before he can use any monitor effectively.

Reviewer: Ralph Tenny

Product Name:	**BUGOUT**
Equip. req'd:	TRS-80C Color Computer with 16K memory
Price:	$19.95
Manufacturer:	Spectral Associates 141 Harvard Avenue Tacoma, Washington 98466

Description: *BUGOUT* is a small (<1800 bytes), well done assembly-language monitor for the Color Computer. Memory modification can be accomplished in two modes: normal examine/modify mode called (S)ubstitute, and (A)SCII, which enters ASCII codes in memory directly from keystrokes. A (F)ill mode allows entry of a given value in each location of a block of memory. The (M)ove memory command allows any memory move, including overlapping code. Two display modes are available: (D)isplay memory in normal screen dump format, and (T)ype, which displays memory contents in ASCII. The cassette handlers are (W)rite, which records assembly language in CLOADM format, and (R)ead, which loads only files made with CSAVEM, W, and CSAVE. In addition, the filename, start address, file type, and last address are displayed. Also, the R command allows an offset to be specified to avoid conflict between programs. Debug and auxiliary commands are also available.

Pluses: Low cost, simple syntax, short learning curve, and convenient, comfortable operation. The S command is an excellent and unique enhancement.

Minuses: No printer output available; no listing furnished. The display scroll speed is too fast, and the V command should be able to verify cassette tapes.

Skill level required: As with any other monitor, the user should have some assembly-language experience. However, this monitor is perhaps the simplest to use and excellent for beginners.

Reviewer: Ralph Tenny

Product Name:	**PAL (Personal Assembly Language)**
Equip. req'd:	Versions for all Commodore computers except the original BASIC 1.0 PET. Printer and Commodore disk or tape optional.
Price:	$50
Manufacturer:	Brad Templeton 271 Westcourt Pl. #201 Waterloo, Ontario N2L 2R8 Canada

Description: *PAL* is a complete 6502 assembler, entirely in machine language. Its syntax is like that of Commodore's own assembler, except that it uses ordinary program files instead of sequential data files. This assembler is used by some of the best known PET programmers.

Pluses: Like its companion program, POWER, *PAL* is extremely simple in concept, flexible in use, and powerful in skilled hands. It occupies only 4K of memory, and may easily be placed in EPROM to leave even more memory free. (It is able to hold its own 4K unassembled source code in a single file.) *PAL* is also extremely fast, assembling itself in under 10 seconds after disk access time. It uses the familiar BASIC screen editor, POWER, or any other programmer's aid, allowing highly flexible editing without learning a new set of commands. *PAL* includes an unassembler, and a converter for MAE assembler source files. The latest version is also able to create relocatable code.

Minuses: Doesn't handle macros, though that skill is planned in an update. BASIC doesn't preserve leading spaces in source files when they are edited. Thus, source files look a bit cramped before assembly.

Documentation: The manual is well printed and quite readable for skilled users. Beginners should probably buy another 6502 machine-language quide to go along with *PAL*'s manual.

Skill level required: Must be familiar with BASIC's screen editor and 6502 assembly language.

Reviewer: James Strasma

Product Name: **Disk Utility Pack**
Equip. req'd: Commodore computer and disk drive
Price: $35
Manufacturer: Dick Immers
152 Brookfield Ct.
Adrian, Michigan 49221

Description: Has three main features: 1. includes a safe and versatile disk monitor called ZAP PET, which is much like Micromon, but for the disk rather than the computer; 2. attempts an automatic disk recovery after otherwise fatal data errors, (such as doing a short "new" to reformat the diskette, or leaving the disk in place while switching disk power off and on); and 3. catalogs up to 99 diskettes.

Pluses: ZAP PET is reliable in use. It has been carefully developed as a doctoral project over the past two years, and has overcome many of the limitations of similar programs. It is also the only such program that uses the hexadecimal and ASCII data used by the better computer monitors. ZAP PET alone is worth the price of the entire package. The disk recovery modules, MAYDAY and EAT MY DISK, are able to restore most or all of the data on diskettes that have been damaged accidentally. The additional data stored by the cataloging programs allows a lost directory to be restored quickly and easily from the catalog.

Minuses: The cataloging modules are slow, compared to competing disk catalogers, due to the extra information they collect. The package needs an expert mode that allows skilled users to avoid the constant query, "Is this information correct?"

Documentation: Six printed pages, well done, but sketchy.

Skill Level Required: Beginners can use most of the package. However, only skilled users should use the edit option of ZAP PET.

Reviewer: James Strasma

Product Name: **Munch-a-bug**
Equip. req'd: Apple II Plus with 48K
Price: $49.95
Manufacturer: Southwestern Data Systems
10159-I Mission George Rd.
Santee, California 92071

Description: *Munch-a-bug* provides a variety of capabilities to aid in debugging assembly-language programs. A few of these are: Trace, a facility for listing 6502 registers as each instruction executes; Breakpoints, a facility to identify paths of execution; Examine, a facility allowing display and change of 6502 registers. There are many other features making the program a truly valuable utility.

Pluses: Assembly-language programming is greatly eased with this product. Large or complex programs are done much faster if intelligent use is made of the product's features. A particularly valuable capability is the Next command which allows tracing but ignores JSRs. A feature for making three back-up copies is built in and works well.

Minuses: A built-in assembler is provided, but it's somewhat primitive. Direct SAVEs of assembly source is impossible. This is, overall, a minor inconvenience; it can be worked around.

Documentation: Excellent. Some fairly original techniques are introduced, but the manual describes them clearly. There is, perhaps, a tendency towards making the examples provided overly simple.

Skill level required: Experienced assembly-language programmer.

Reviewer: Chris Williams

Product Name: **OS-96**
Equip. req'd: CBM 8032 with 8096 expansion memory board and Commodore disk
Price: $200
Manufacturer: A.B. Computers
252 Bethlehem Pike
Colmar, Pennsylvania

Description: *OS-96* is an extended BASIC interpreter. It expands BASIC workspace to an apparent 64K free (32K reserved for program text, and 32K for variables.) Also adds several commands and enhancements to BASIC, including most of those in the SM KIT programmer's aid. Further enhancements to the interpreter are planned later.

Pluses: Leaves *lots* of memory space free for large programs. Changing program lines, or chaining large modules behind small ones, doesn't clear or overwrite variables. The BASIC stack has been lengthened, allowing deeper nesting of loops and subroutines. Key sequences access the 8032's extended editing commands. Program lines and variable data may now be up to 255 bytes long. Spaces are automatically deleted from program lines, and reinserted on LISTing, with a further option moving each statement to a new line, properly indented for structure. *OS-96* is fully compatible with PET BASIC syntax and programs.

Minuses: Program load addresses differ from usual BASIC. Includes a "load at" command to help. The AUTO line number feature often turns on when not wanted. None of the usual SYS calls or enhancement ROMs work with *OS-96*.

Documentation: The instructions I saw were preliminary. I found them adequate for those who already know PET BASIC, though they hint at undocumented commands.

Skill level required: Beginner to expert.

Reviewer: James Strasma

Product Name:	**The WIZ Database Manager**
Equip. req'd:	CBM 8032 or 8096 computer with 8050 disk drive. CBM or ASCII printer optional. Uses phantom ROM at $9000.
Price:	$495
Manufacturer:	Dr. Daley Software Water Street Darby, Montana 59329

Description: *WIZ* is a data manager package. It is limited to a single file, but the file may use up to 30 diskettes, each with 2,000 or more records. Each record may be up to 250 characters long, and contain up to 64 fields of up to 78 characters each.

Pluses: Handles vastly larger numbers of records than competing programs. Also includes built-in plotting. Like its best competitors, the *WIZ* uses CBM's recommended screen input editor, has programmable arithmetic, and handles simple reports. It also allows sorting on any field or combination of fields, and either fixed field or global searches, with or without regard to the case of the text. Results may be sent to screen, printer, or Word-Pro sequential disk file. Includes listing to allow reading and writing its data from other programs, and can jump from itself to other user programs. Better yet, it can read in data from sequential disk files, avoiding retyping when upgrading from a lesser data package.

Minuses: Moderate attempts are made to keep the user from halting, listing, or changing the programs. The phantom ROM will activate outside the *WIZ*, if its last two addresses are PEEKed; whether this is a problem depends on what other ROM shares the socket.

Documentation: Self-documenting. Includes a thick and nicely packaged manual, but you may not need it. The *WIZ* explains itself merely by typing "help". This works virtually anywhere in the program, providing a total of 125 help screens.

Skill level required: Even beginners should be able to use this package effectively.

Reviewer: James Strasma

Product Name:	**Write Away**
Equip. req'd:	Apple II or Apple II Plus, 48K, 1 disk drive. Recommend modified shift key, lower-case adapter, 80-column card, second disk drive.
Price:	$175.00
Manufacturer:	Midwest Software Assoc. P.O. Box 301 St. Ann, MO 63074 1-800-835-2246, Ext. 467

Description: A versatile word processor providing the user with a variety of levels of capability. Terminology and procedures are quite similar to DEC's PDP word processing software.

Pluses: The expandability is excellent. If one is familiar with DEC procedures, learning time is near zero.

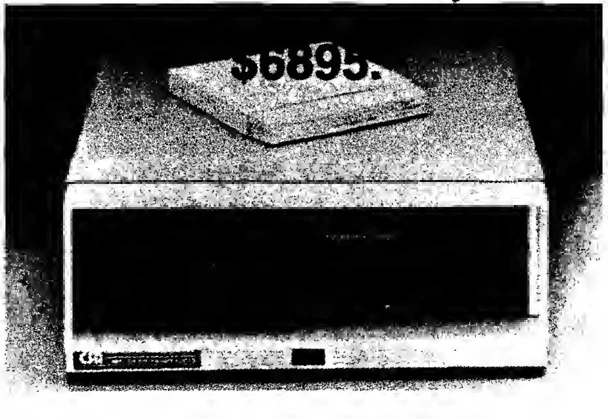
Minuses: Very little can be found wrong with this program. There is a lengthy tutorial on the program disk that, perhaps, takes longer than it should to complete.

Documentation: Excellent. The booklet provided is concise and informative. A separate quick-reference card of commands is included in the package. Without it, the user would have to occasionally erase his screen of text in order to use the HELP command.

Skill level: No programming language knowledge needed. Some general computer familiarity is desirable.

Reviewer: Chris Williams

(Continued)



Product Name: **Amper-Magic**
Equip. req'd: 32K Apple II or Apple II Plus; Applesoft in ROM or Language Card and one Disk II with DOS 3.3.
Price: $75.00
Manufacturer: Aurora Systems, Inc.
2040 E. Washington Ave.
Madison, WI 53704
Author: Bob Nacon

Description: A menu-driven utility program that allows machine-language subroutines to be added to Applesoft programs. Each subroutine is given a name (see below) and is callable from the BASIC program using the Applesoft Ampersand (&) command. Twenty-three ready-to-go subroutines and their descriptions are included with the package. Volume II, with 26 routines, will be released by the time you read this.

Pluses: *Amper-Magic* provides a convenient way to package ampersand routines and Applesoft programs together. This allows the user to avoid extra BLOAD commands and possible adjustment of program pointers when setting up a program that uses ampersand subroutines. Many options are provided for ease of use. For example, any table of subroutines created for use by one program may be saved in a binary file and used as a unit in any other program desired.

Minuses: The names given to the subroutines are restricted to four bytes. A byte may be either a single ASCII character or an Applesoft token name (e.g., GOTO, NEXT, PRINT, etc.). This restricts the user to unmnemonic names if only ASCII characters are used.

Documentation: Good. A quick reference card summarizes system usage.

Skill level required: Facility with Applesoft. No knowledge of machine language is needed.

Reviewer: Richard C. Vile, Jr.

Product Name: **Apple Flasher**
Equip. req'd: Apple II with Applesoft in ROM or RAM card. One or more disk drives with DOS 3.3.
Price: $34.40 plus $1 for shipping
Distributor: Crow Ridge Associates, Inc.
P.O. Box 90
New Scotland, NY 12127
(518) 765-3620

Description: *Apple Flasher* provides a convenient, rapid way to find and display high-resolution pictures that have been previously stored on diskettes by other Apple II programs. It loads the names of assumed picture files into the program and offers four options for their display: 1. A single keystroke to present picture; 2. Each picture scanned for about two seconds in rapid succession; 3. Picture files displayed in an emulation of a manual slide drum slide projector; 4. Automatic projector mode. Back-up replacement is available for $8.50 plus return of the damaged disk. One back-up copy will be sold in addition to the master disk for this price.

Pluses: The author has developed a very fast file recovery algorithm that loads frames in about two seconds. You can also load three files at once with instantaneous shuttle to either hi-res screen one or two. Files may be on two drives and "punched up" in any sequence.

Minuses: The program does not allow for reordering the presentation of the files differently from the order they are saved on the disk. The price of the program seems somewhat high in view of its limited function. Without carrying out the task of entering the projection times in each image file, the auto-display function is of limited use.

Reviewer: David P. Allen

Product Name: **Color Scribe**
Equip. req'd: TRS-80 Color Computer, 32K with disk.
Price: $49.95
Manufacturer: Computerware
Box 668
Encinitas, CA 92024

Description: This full-featured text editor/formatter includes full disk and cassette ASCII-file compatibiity. The text editor is line-oriented, and can be used to edit BASIC files, as well as many other text applications. The editor includes edit, change, add, copy, move, and replace. The formatter is very powerful and automatically justifies left and right text. Headers and footers are allowed, as is underlining and centering. Indentation and sending of printer control codes are accepted. User remarks can be entered into files, and files larger than memory are easy to create.

Pluses: Easy to learn for simple applications, yet offers more powerful functions when required. The formatter is one of the best I have seen. The disk operating system works well now since bugs from an earlier version of the program have been corrected in this version. Files can be as large as a disk can hold by using the 'MORE' command. The READ command brings text from a file on disk into any desired position in the file in memory. Files are backed up automatically as they are loaded. A MACRO command allows the user to specify single-key entry of one or a series of commands. ESCape and CHR$ commands can be specified by the formatter to allow you to use special print options, change character fonts, etc., in the printer. Commands can be entered in either upper or lower case.

Minuses: Since the program is line oriented, the user must press the ENTER key at least every 127 characters. Text is difficult to visualize on screen, and the screen print formatter does not use a "window" format to view the text. As a result, you must do a printout to actually view the result of formatting commands. When working on a file brought in from disk, the file is open. If you shut off the computer, or exit Scribe *via* Reset, or other unauthorized means, you will have an open file on the disk. You will probably not be able to read this file again. It is possible to end up in BASIC if you press the BREAK key while accessing the directory. The manual does warn of this. A minor bug does not allow using the slash bar as extension delimiter; a period is used as delimiter.

Documentation: The 40-page operation manual is of excellent quality. A demonstration file is included on the disk that assists in understanding the operation of the program.

Skill level required: The progam operation is easy to learn, even for the novice operator.

Reviewer: John Steiner

Commodore Programs Move into the *Fast Lane* with PETSPEED

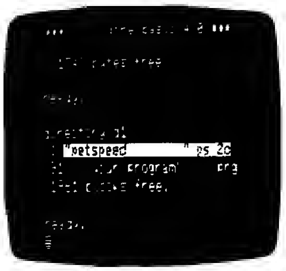
Petspeed — **The Optimizing Basic Compiler that runs Commodore BASIC 40 times faster. You can dramatically reduce long processing times, tedious disk handling, and long print runs. No other compiler can offer the same speed, compatability and trouble-free compiling as Petspeed.**

Compatible —Petspeed compiles any BASIC application and is available for any combination of 4000 and 8000 series Commodore equipment.

Faster The key that sets Petspeed apart from other compilers is **optimization.** Where most compilers merely translate from one language to another, Petspeed analyzes your source program to eliminate unnecessary complexities, thus speeding processing time. Your program is reduced to much smaller components and reassembled into a far more efficient form. Compare these optimizing features:

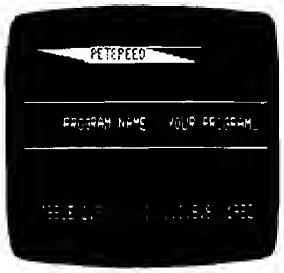
- 4-Pass compiler.
- Automatically uses faster integer arithmetic when possible.
- Automatically handles frequently occurring variables and arrays.
- Subroutines no longer need be located at the beginning of your program.
- Petspeed automatically calls all subroutines at maximum speed.
- Petspeed runs twice as fast as other compilers.
- Larger programs require far less memory when compiled.

Easy to Use Petspeed is as easy to use as these screen displays illustrate.



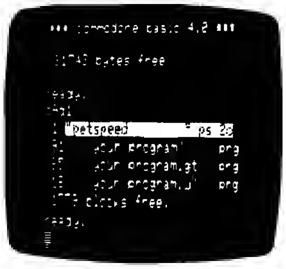
```
***          [illegible] 4.2 ***

 [illegible] bytes free

ready.

directory d1
 "petspeed          " ps 2c
[illegible]    your program'      prg
[illegible] blocks free.

ready.
```

Directory BEFORE compilotion.



```
      PETSPEED

   PROGRAM NAME   YOUR PROGRAM_

[illegible]
```

Simply type in your program name.



```
*** commodore basic 4.2 ***

 [illegible] bytes free

ready.
[illegible]
 "petspeed          " ps 2c
[illegible]    your program'      prg
[illegible]    your program.gt    prg
[illegible]    your program.u'    prg
[illegible] blocks free.
ready.
```

Directory AFTER compilation

It isn't necessary to add compiler directives. Simply type in the program name. In less than 2 minutes, you'll see your program run up to 40 times faster.

Security A security device is provided to run Petspeed, but no runtime key is necessary for compiled programs. You're free to build in your own protection. Petspeed code cannot be listed by others, so compiled programs cannot be tampered with. Your programs belong entirely to you.

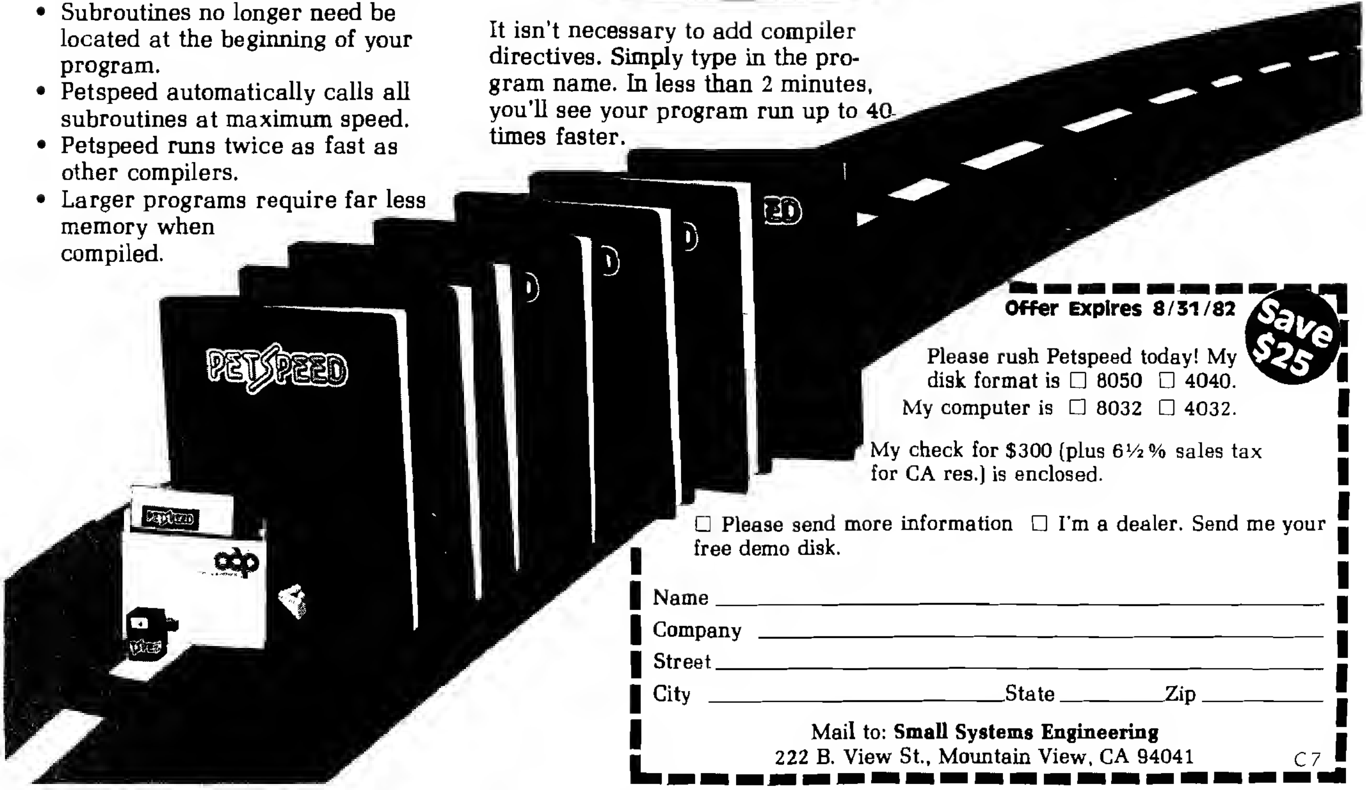
Move your Commodore into the fast lane today with Petspeed. And save $25 too! Send this coupon today to

SSE

SMALL SYSTEMS ENGINEERING
222 B View Street
Mountain View, CA 94041
(415) 964-8201

Dealers: Ask for our free demo disk.

Price $325



Offer Expires 8/31/82

Save $25

Please rush Petspeed today! My disk format is □ 8050 □ 4040.
My computer is □ 8032 □ 4032.

My check for $300 (plus 6½% sales tax for CA res.) is enclosed.

□ Please send more information □ I'm a dealer. Send me your free demo disk.

Name ____________________
Company ____________________
Street ____________________
City ____________ State ________ Zip ________

Mail to: **Small Systems Engineering**
222 B. View St., Mountain View, CA 94041

C7

Dr. William R. Dial
438 Roslyn Avenue
Akron, OH 44320

6502 Bibliography

1. Commodore Magazine (April/May, 1982)

Scott, Dave, "An Easy Cursor Positioning Routine," pg. 82-85.

A machine-language program for the Commodore Microcomputers which allows printing a message at a specific x,y location on the screen. It is entered *via* the "SYS" command.

2. MICRO No. 48 (May, 1982)

Engstrom, Christer, "BASIC to Machine-Language Interface," pg. 13-17.

Since the AIM lacks a CALL or SYS function, it is difficult to communicate between BASIC and machine language. This interface routine makes possible the entry of the machine-language address through a BASIC variable.

3. Compute! 4, No. 5, Issue 24 (May, 1982)

Berlin, Stanley M., "Putting the Squeeze on Your VIC-20: Getting the Most Out of 5000 Bytes," pg. 136-144.

When you write a program so large that it will need every last memory cell in your computer, this article shows several ways to reduce the size of your program. In general, the suggestions may apply to other machines using BASIC.

4. The Commodore Transactor 3, Issue No. 5 (May, 1982)

Hook, David A., "VIC Loader for PET/CBM," pg. 41-48

This procedure, BASIC program, and machine-language routine will make it possible for VIC-20 users to exchange tapes with PET/CBM users. These routines accommodate several possibilities of incompatibility.

5. Antic 1, No. 1 (April, 1982)

Capparell, James, "Systems Guide: Memory Map — Page 0," pg. 9-11, 15.

In the next few issues of this new publication serving the Atari micros, a complete memory map will be constructed giving the reserved memory locations in RAM and ROM with a description of the purpose of each location. The first 128 of page zero are reported in the current issue.

6. PEEK(65) 3, No. 5 (May, 1982)

Eidbo, Carl, "OS-65U Disassembly Aid," pg. 3-5.

The unavailability of an OS-65U disassembly manual led to a project to disassemble specific areas. Both BASIC and machine-language versions of a disassembler are given for OSI micros.

7. Creative Computing 8, No. 5 (May, 1982)

Ott, Jack P., "Listing/Copying Apple Text Files," pg. 144-154.

The lack of a simple command to list or copy text files on the Apple prompted design of a procedure to do this. Written in BASIC, simple editing of whole records is made possible, but is not as versatile as a true text editor.

8. Mini'App'Les 5, No. 5 (May, 1982)

Murrell, Mike, "Cursor Menu," pg. 9-10.

A cursor-oriented menu control program for the Apple. Written in Applesoft.

9. Call —A.P.P.L.E. 5, No. 5 (May, 1982)

So, Edward C., "Picture Compression," pg. 21-35.

A program in machine language for the Apple which provides a method of storing and retrieving an Apple hi-res picture in less than the 34 sectors it normally requires. The program consists of five separate routines. By examining for duplicated pixels within the picture, one can use less than 960 pixels to store the picture.

10. Byte 7, No. 5 (May, 1982)

Sweet, Jerry N., "Chedit," pg. 426-444.

A graphics-character editor enabling the Apple Pascal user to define his own character set, using the Turtlegraphics system.

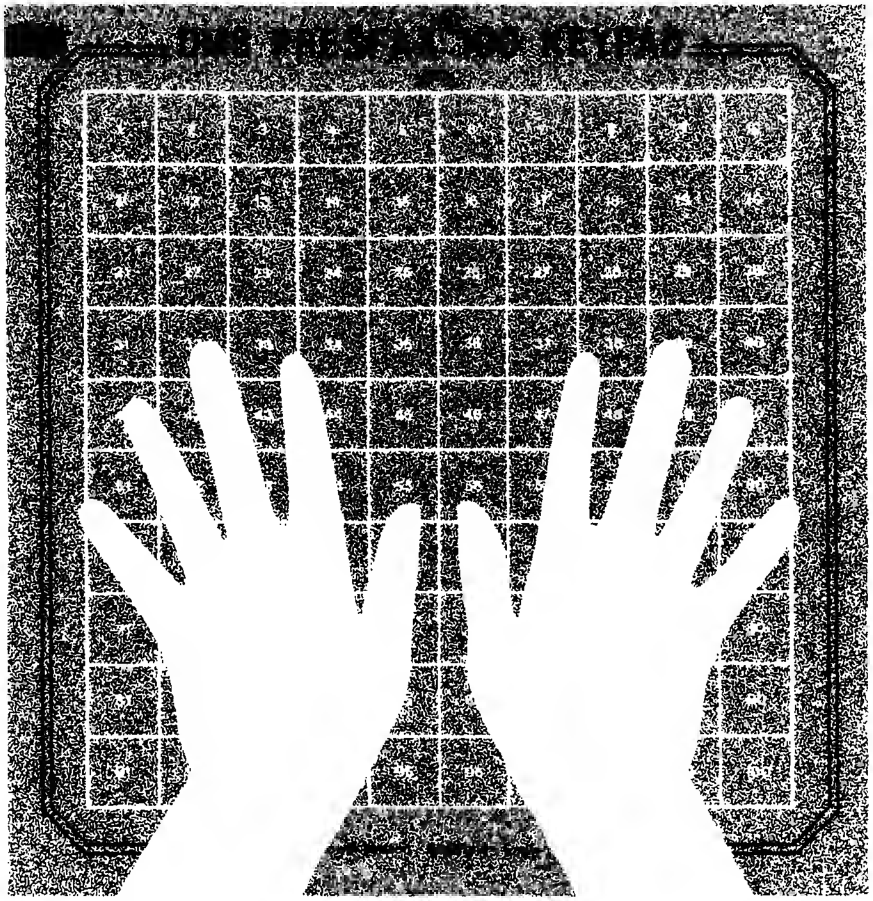
Microcomputer Information Resources

MICRO wants to make sure our readers are aware of the excellent sources of microcomputer bibliographic information that are available.

- *Microcomputer Index* is a periodical that provides a subject index for a cross-section of popular microcomputer magazines. It includes abstracts. Published by Microcomputer Information Services, 2646 El Camino Real #247, Santa Clara, CA 95051, *Microcomputer Index* has put more than 10,000 articles, indexed and abstracted from 23 periodicals, on line with Lockheed's DIALOG service. *The Index* is probably the best single source of bibliographic information in book form about articles published in microcomputer magazines. Compiled by W.H. Wallace and published by Missouri Indexing, Inc. (P.O. Box 301, St. Ann, MO 63074), *The Index* is not only comprehensive, but so well organized that using it is a pleasure.
- *Micro ... Publications In Review* is another helpful publication (Vogeler Publishing Inc., P.O. Box 489, Arlington Heights, IL 60006). This periodical reproduces the tables of contents of the latest issues of the major microcomputer magazines, and provides a subject index.

HANDS-ON LEARNING

Presfax® and Apple® II



Young students, the handicapped and many others are finding the Apple® II computer more accessible than ever before. Thanks to the Presfax 100® Touch Teacher keypad from Computer Data Services.

With a square, touch-sensitive area consisting of 100 "keys," the Presfax 100 keypad fits right over the Apple II's keyboard, freeing users from any restrictions they may encounter with the standard keyboard arrangement.

The keypad grid can be covered with overlays composed of whatever symbols the user feels most comfortable with — large alpha-numeric characters, braille, special symbols and pictures, etc. Grid values and coordinates are memory-stored and assigned individual audible pitches to ensure user feedback.

Driver software comes on a 3.3 DOS disc complete with full instructions on how to incorporate it with the purchaser's own programs. Demonstration and educational programs are also available. Contact Computer Data Services, P.O. Box 696, Amherst, N. H. 03031 for full details. Or call (603) 673-7375.

Suggested Retail Price	$175.00
Shipping	3.50

Dealer Inquiries Invited

CDS™

COMPUTER DATA SERVICES
P.O. Box 696 Route 122
Amherst, NH 03031
(603) 673-7375

MasterCard VISA

Dr. William R. Dial
438 Roslyn Avenue
Akron, OH 44320

6809 Bibliography

64. BYTE 7, No. 5 (May, 1982)

Kocher, Christopher P. and Keith, Michael, "Six Personal Computers from Japan," pg. 61-102.
Descriptions in detail of six Japanese computers including the 6809-based Canon CX-1, the Hitachi MB-6890, and the Fujitsu FM-8.

Barden, William, Jr., "Ports of Entry and Soft Breezes for the Color Computer and Model III," pg. 162-198.
Instructional article for the Radio Shack 6809-based Color Computer. A $10 anemometer and other remote-sensing projects using the cassette interface.

65. MICRO No. 49 (June, 1982)

Tripp, Robert M., "Editorial," pg.5.
Discussion of new developments relating to the 6809 microprocessor.

Clements, William C., Jr., "Add a VIA and Speech Synthesizer to the Color Computer," pg. 19-21.
Hardware article for the TRS-80 Color Computer, adding two user-accessible, 8-bit bidirectional I/O ports to your computer and interfacing an inexpensive speech synthesizer.

Dial, William R., "Resource Update," pg. 99-100, 110.
A list of magazines which contain information about the 6502/6809 microprocessors on a reasonably regular basis.

Staff, "Software Catalog," pg. 101-109.
Description of some 50 different software items, including a number for 6809-based systems.

Staff, "Hardware Catalog," pg. 111-113.
Some 20-odd items of hardware are described including several for 6809 systems.

Dial, William R., "6809 Bibliography," pg. 117.
A number of literature references related to 6809-based systems.

66. Apple Assembly Lines 2, Issue 9 (June, 1982)

Sander-Cederlof, Bob, "Implementing New Opcodes Using 'BRK'," pg. 2-5.
How to use the 6502 to simulate neat instructions from the 6809 like LEAX, BSR, BRA, etc.

67. Microcomputing 6, No. 6, Issue 66 (June, 1982)

Staff, "C Development Tools for the MC6809," pg. 162.
An announcement of Introl-C, a set of software for 6809-based C programs, including a C compiler, a 6809 assembler, an object code linker, and an object code library manager.

68. BYTE 7, No. 6 (June, 1982)

Barden, William, Jr., "A General-Purpose I/O Board for the Color Computer," pg. 260-282.
This hardware article describes a low-cost interface which can be plugged into the ROM slot of the Radio Shack TRS-80 Color Computer.

69. The Rainbow 1, No. 12 (June, 1982)

Lewandowski, Dennis S., "The Assembly Corner," pg. 6-7.
Discussion of indexed addressing for the Radio Shack TRS-80 Color Computer, also designated the 80C or CoCo. Several other instructions for this micro are also discussed.

Waclo, John, "80C Picks the National Football League," pg. 12-18.
Keep track of how the NFL is doing with this program for the TRS-80 Color Computer.

Shorter, Ted, "Ice Lander Can Be A Slick Run," pg. 25-27.
A lunar lander type of game for the 80C.

Nolen, Bill, "Now a Whole Bag of Dice," pg. 38-40.
How to replace your $5.00 set of dice with a $600 computer, with a few extras to boot. For the TRS-80 Color Computer.

Curtis, H. Allen, "Getting More from PMODE 4," pg. 42-46.
Discussion of color graphics on the 80C demonstrating that PMODE 4 has six other color sets. Several short programs are given.

Scerbo, Fred B., Dobbert, Robert, and Haggerty, Dale, "Swamp Wars Can Win Your Croix de Lillypad," pg. 49-50.
A color game for the TRS-80 Color Computer.

Mir, Jorge, "A Data File to Store All Your Information," pg. 57-62.
A flexible file system for the Color Computer, with search function, sorting, etc.

Moses, Ray, "Stars Fall on 80C," pg. 64-65.
A program giving a display of the sky using a variety of times of the year.

70. '68' Micro Journal 4 No. 5 (May, 1982)

Anderson, Ronald W., "Flex Uer Notes," pg. 10-12.
Discussion of the use of Pascal including a dump utility for 6809 based systems.

Nay, Robert L., "Color User Notes," pg. 13-14.
Notes on the Super Color Writer and Flex9 for the Radio Shack 6809-based Color Computer.

Cadmus, Ray, "OS-9 Notes," pg. 15-16.
Notes on the OS-9 operating system for the 6809 including a program HDIR, a hiarchial directory.

Rowley, Blair A., "Uniflex and RSTS," pg. 19-22.
Intercommunication between Uniflex and RSTS developed in 6809 assembly language.

Lilly, Randal, "Transfer" pg. 22-24.
A computer link program that allows a main computer (SWTCP 6809) to transfer and run programs on a remote CPU (6800 or 6809 computer).

Jones, Bob, "Othello," pg. 34-35.
Othello for 6809 based systems.

Preble, Laurence D., "Error Messages," pg. 37.
A patch for Version 17 of TSC's Extended BASIC for the 6809 to better handle error messages.

Liu, Bruce, "Overlay Systems," pg. 37-39.
Improved Pascal routine for 6809 based systems.

71. 80 Micro Issue No. 30 (June/July, 1982)

Berenbon, Howard, "SDS80C Editor/Assembler/Monitor" pg. 70-71.
Review of a 6809 assembler/editor/monitor.

Commander, Jake, "Spiromania-Part II," pg. 106-115.
Machine code routine for color graphics on 6809 based systems including the Radio Shack Color Computer.

Norman, Scott L., "Color Computer Utilities," pg. 116-120.
Power Pack and Color Diagnostics, Editor and Assembler for the Radio Shack Color Computer are reviewed.

Leichtman, Kerry, "Bob Rosen — A Colorful Success Story," pg. 174-175.
Build a better bulletin board and the world will beat a path to your door. Using the 6809-based Color Computer.

Miller, Franklyn D., "Extended Color Basic," pg. 266-270.
Discussion and hints for using this system for the Color Computer.

MICRO™ Software Catalog

Name: **Financial Records Manager**
System: Apple II, DOS 3.3
Memory: 48K
Language: Applesoft and Machine
Hardware: Disk drive (printer optional)
Description: A menu-driven, user-friendly program to collect, sort, store, and analyze up to 750 checking/budget records and 500 tax records. Includes 30 user-defined budget categories and 50 user-defined tax categories, automatic payments, search and edit functions, average and percentage budget analysis, tax summary (including approximate taxes due), "What if?" analysis, and utilities.
Price: $44.95
Includes typeset manual, program disk (copyable).
Author: D.M. Whittaker
Available:
Cybertech
1500 W. Shaw, Suite 404
Fresno, CA 93711
(209) 222-9094

Name: **How to Operate the Apple II Plus**
System: Apple II Plus
Hardware: One disk drive, standard audio cassette player
Description: An audio cassette mini-course for the first-time computer user. Three audio cassettes and Apple's own DOS 3.3 System Master diskette are used to teach the essentials of computer operation: how to run programs, enter data, save and organize files, make backup copies, modify programs, and much more. No programming or technical knowledge assumed.
Price: $49.95
Includes three spoken-voice cassettes with 28-page Operator's Guide in a vinyl binder.
Author: Howard Manthei and Lee McFadden
Available:
Fliptrack Training Tapes
A Division of Mosaic Media
P.O. Box 711
Glen Ellyn, IL 60137
(312) 790-1117

Name: **Bowling Data System 2.0**
System: Apple II
Memory: 48K minimum
Language: Applesoft ROM
Hardware: 1 disk drive (2 recommended), DOS 3.3, 80-column printer
Description: A database program for bowling league secretaries. Stores and computes data for multiple (and/or mixed) leagues of up to 40 teams, each with up to six bowlers per team (not including substitutes). The reports generated are: weekly recap, score sheet, season average, and team listing. Many options are included. For each team a cumulative record is kept of all pertinent data.
Price: $149.95
Includes documentation, one diskette.
Author: Arnold Hooton, Published by Rainbow Computing, Inc.
Available:
RCI Marketing
19517 Business Center Dr.
Northridge, CA 91324
(213) 349-0300

Name: **The UCSD p-System™**
System: The UCSD p-System
Memory: 48Kb runtime environment; 64Kb development environment.
Language: Written in UCSD Pascal and Assembly
Support: UCSD Pascal, FORTRAN-77, BASIC, and Assembly
Hardware: 8086, Z80, 8080, 8085, 6502, 9900, LSI-11/PDP-11, and 68000
Description: *The UCSD p-System* is a stand-alone operating system for most microcomputers based on the 8086, 8080, Z80, 6502, 6809, 9900, 68000, and LSI-11/PDP-11 microprocessors. *The UCSD p-System* includes an operating system, file handler, screen-oriented editor, line-oriented editor, macro assembler, interpreter, and utilities.
Price: $375.00
Includes object code for the UCSD p-System, UCSD p-System User Manual, Architecture Guide, Installation Guide, and Beginner's Guide to the UCSD Pascal System, with tutorial disk.
Available:
SofTech Microsystems, Inc.
9494 Black Mountain Rd.
San Diego, CA 92126
(714) 578-6105

Name: **Waterloo microFORTRAN**
System: Commodore SuperPET, Volker-Craig 2900, 3900, 4900, Northern Digital microWAT
Description: *Waterloo microFORTRAN* is a special dialect designed for teaching purposes. It has many of the characteristics and much of the flavour of normal FORTRAN, but varies significantly from established standards for that language. This language processor has many of the important characteristics of the WATFIV-S compiler which is widely used on IBM computers, plus some features from the new FORTRAN-77 definition. Examples of language features supported are FORMAT, subroutines and functions, multi-dimensional arrays, extended character-string manipulation, structured programming control, and file input/output. In addition, the interpreter provides a powerful interactive debugging facility.
Available:
Waterloo Computing Systems Limited
158 University Ave. West
Waterloo, Ontario
Canada N2L 3E9

Name: **Storm**
System: Radio Shack Color Computer
Memory: 16K
Language: Assembly Language
Hardware: Joysticks
Description: *Storm* brings the excitement of the arcade to your home. A tempest of a game, it has 135 levels of play, fast action, graphics, and sound. It challenges the very best arcade experts and offers fun for the beginners too. There is no other high-resolution graphics arcade game like it!
Price: $24.95 on cassette
$29.95 on disk
Includes program and instruction manual
Available:
Computerware
Box 668
Encinitas, CA 92024
(714) 436-3512

Name: **FORTH-79 Version 2**
System: Apple II or Apple II Plus (also Z-80 CP/M, NorthStar Advantage, and NorthStar DOS
Memory: 48K
Language: Machine Language and FORTH-79
Hardware: 1-14 disk drives (13- or 16-sector-compatible)
Description: *FORTH-79 (Ver. 2)* is a structured language suited for systems and applications programming where execution speed is important. It meets all provisions of the FORTH-79. Standard and programs are portable, faster than BASIC, and provide an interactive environment. Its high-speed compiled code is well suited for real-time applications. Includes screen editor, macro-assembler, strings, 32-bit integer arithmetic. Floating point and hi-res Turtle Graphics are also available.
Price: $99.95 - BASIC
$139.95 - hi-res and floating point
Includes software and 200-page manual (including FORTH-79 and fig-FORTH).
Author: Martin Tracy and Philip Wasson
Available:
MicroMotion
12077 Wilshire Blvd.
Los Angeles, CA 90025
(213) 821-4340

(continued)

Software Catalog *(continued)*

Name: **BASIC Compiler™**
System: The UCSD p-System
Memory: 48Kb runtime environment; 64Kb development environment
Language: BASIC
Hardware: 8086, Z80, 8080, 8085, 6502, 9900, 6809, 68000, LSI-11/PDP-11

Description: The *BASIC Compiler* is a powerful and flexible BASIC offering you: expanded syntax, including IF THEN ELSE construct; INCLUDE files; and virtual arrays. The *BASIC Compiler* is fully integrated and compatible with UCSD Pascal™ and FORTRAN-77.

Price: $225.00
Includes object code for the BASIC Compiler and BASIC Reference Manual.

Available:
SofTech Microsystems, Inc.
9494 Black Mountain Rd.
San Diego, CA 92126
(714) 578-6105

Name: **Cash+Plus**
System: Apple II Plus with DOS 3.3
Memory: 48K
Language: Applesoft BASIC

Description: Point-of-sale program with inventory controller and management reporter. Can also be used with optional cash drawer.

Price: $395.00
Includes programs and manual.

Author: Lynn Graybiel

Available:
Southeast Computer Concepts Corp.
500 E. Spanish River Blvd.
Boca Raton, FL 33431

Name: **Bond Yielder**
System: Apple II Plus
Memory: 48K
Language: Applesoft
Hardware: Disk drive, optional printer

Description: *Bond Yielder* performs fast and accurate calculations on the profitability of fixed income investments. This package can replace the $1200 desk-top calculator now standard in the investment industry.

Price: $149.95

Author: Donald Brown

Available:
CE Software
801 73rd Street
Des Moines, IA 50312

Name: **TRA/C1**
System: OSI Superboard II, C1P
Memory: 4K or more
Language: BASIC and 6502 ASM
Hardware: Cassette tape

Description: *TRA/C1* single steps and traces machine-code programs and displays all registers, processor status, stack pointer, program counter, instruction and operand. Does not disturb any registers or the stack. This program is an essential aid to OSI machine-code or asssembly-code writers.

Price: $17.95
Includes casette tape and instructions.

Available:
D. Wolf, Ph.D.
Box 565
Port Hueneme, CA 93041

Name: **Universal Mailing List**
System: Any Commodore 6502 or 64
Memory: 3.5 (Vic 20) — 96K on the CBM
Language: BASIC
Hardware: Disk drive or tape drive

Description: The program allows you to change, view and update addresses. It allows you to print one or all addresses onto mailing labels. It will build a list and save it onto a cassette tape for recall. The program is designed for the VIC 20, but it runs on the 4016, 4032 and the 8032. Capacity ranges from 21 addresses on 3.5k to 359 addresses on 32k.

Price: $29.95
Includes 3 ring binder, users' manual and a cassette page that holds six cassettes for the program and data files.

Author: S. Mark Vanderbilt

Available:
Computer Barn
Old Town Salinas
319 Main Street #2
Salinas, CA 93901
(408) 757-0788

(Continued)

Name: **Softerm**

System: Apple II or Apple II Plus

Memory: 48K, Autostart ROM

Language: Assembly - 6502

Hardware: Serial communications, optional 80-column video, optional printer

Description: High-speed CRT emulator up to 9600 baud for following terminals: IBM 3101 series, DEC VT-100 and VT-52, Data General D-200, ADDS Regent Series, Lear Siegler ADM-3A and ADM-5, Hazeltine 1400 and 1500 series, TeleVideo 900 series. Also includes file transfer with data compression, keyboard macros, automatic dialer with built-in phone book, local connect or remote operation using standard modems.

Price: $150.00
Includes progam diskette, configuration diskette, manual.

Available:
Softronics
6626 Prince Edward Place
Memphis, TN 38119
(901) 755-5006

Name: **FLEX - OS9 Copy Utility Program**

Description: *Metacopy 130* is a file-copy utility that transfers any type of OS9 file to a FLEX sequential file, or FLEX sequential files to an OS9 file. This utility also provides a directory of FLEX disk while in the OS9 operating system. The program is menu driven for ease of use.

Price: $135.00

Available:
Meta Lab
P.O. Box 1559
Suite 106
2888 Bluff St.
Boulder, CO 80301
(303) 499-4236

Name: **Mojave Desert Adventure**

System: VIC-20

Memory: 3.5K

Language: BASIC

Hardware: Cassette, joystick (optional)

Description: Explore the Mojave Desert in search for gold. You find gold but trouble finds you. Crunched and randomized programming makes this treasure hunt last hours. Game is over when you find all the gold or die of thirst. Random placement of key objects makes this a new adventure each time played.

Price: $9.95
Includes cassette tape, instructions.

Author: Dennis McCormack

Available:
The Byte House
Box 981
Salem, NH 03079

Name: **Roterm**

System: OSI Superboard II, C1P

Memory: 8K

Language: BASIC and 6502 Assembly

Hardware: Modem

Description: *Roterm* turns the computer into a high performance cipher terminal with special enhancements for convenient encryption and decryption of electronic mail. *Roterm* emulates a classic rotor cipher machine and can operate at continuous data rates of 300 baud. *Roterm* is also available specially configured for Cegmon users or users of the new model Superboard.

Available:
D. Wolf Ph.D.
Box 565
Port Hueneme, CA 93041

Name: **CW Morse**

System: Atari 400 or 800 VIC-20 Computers

Memory: 16K - Atari
5K VIC (expansion memory optional)

Hardware: Two transistor, 1 IC interface

Description: *CW Morse* allows your computer to become a Morse terminal for an amateur radio station. It is capable of sending and receiving Morse code at speeds of up to 25 wpm. Includes multiple 255 character message buffers, special function keys, typeahead keyboard buffer, and automatic speed control on receive. Package includes documentation, interface schematic, and I/O connector. Send SASE and type of computer for more information on this and other software available.

Price: $19.95
Includes cassette.

Available:
Kinetic Designs
401 Monument Rd. #171
Jacksonville, FL 32211

Name: **High-Resolution Plotting Routine**

System: VIC 20

Memory: 2K

Language: BASIC

Hardware: 1515 printer, tape player

Description: This routine turns the VIC 1515 printer into a plotter with the highest possible resolution (72 dots/inch horizontally and vertically). Will plot the results of user-supplied programs, functions, or data logging routines. Size of the plot and automatic scaling with number labels and tic marks are continuously user-adjustable. Length of the plot may be infinite, allowing strip chart recording.

Price: $10.00
Includes cassette and instructions

Author: Tim Bowker

Available:
Scientific Software
525 Lohnes Dr.
Fairborn, OH 45324

Name: **Legend Slide Select**

System: Apple II

Memory: 48K

Language: Assembly

Hardware: Legend 64K or 128KDE Card

Description: *Legend Slide Select* offers a way to control the display of hi-res pictures. The program permits the Apple II to act as a slide projector, and allows the user to flip forward or backward. Control of the displayed pictures is user-selectable and can be controlled from the keyboard, game paddles, or set to run automatically.

Price: Free with card
($349 retail)
Includes program and manual.

Available:
Legend Industries
2220 Scott Lake Road
Pontiac, MI 48054
(313) 674-0953

(continued)

Name: **Access**
System: Apple II Plus
Memory: 48K
Language: Applesoft, Machine Code
Description: *Access* is a general-purpose database manager. Special features include: word processor-style editor (delete/insert characters, etc.); free format screen design; supports most 40- or 80-column upper- and lowercase hardware modifications; record retrieval time by index less than three seconds and using any search parameters less than 23 seconds; up to 1521 characters per record; up to 39 fields per record; up to 39 characters per field; up to 20 calculated fields per record; short forms option, and much, much more.
Price: $320.00
Includes manual, two disks.
Author: Nik Spicer and Mike Slaughter
Available:
Spider Software Ltd.
98 Avondale Road
South Croydon
Surrey, England
Tel. 01-680 0267

Name: **SBCS AGRI-LEDGER**
System: Apple II or Apple II+, DOS 3.3
Memory: 48K
Language: Applesoft
Hardware: 2 disk drives, 80 column printer
Description: Organize farm financial records with the SBCS Agri-Ledger! It is Menu driven, a user oriented program with clear, easy to follow documentation. Numerous reports, budgeting, large flexible chart of accounts, yearly history, enterprise analysis, current market valuation can be done. Warranty, free program updates, and friendly customer service are provided.
Price: $395.00
Includes program disk, program backup, data disk and documentation. Demo available for $30.
Author: David McFarling
Available:
Small Business Computer Systems
4140 Greenwood
Lincoln, NE 68504
(402) 467-1878

Name: **Strobe Business Graphics**
System: Apple DOS 3.3 CBM DOS
Language: BASIC
Hardware: Apple II, Apple III, or CBM/PET with Strobe 100 Plotter
Description: *Strobe Business Graphics* provides excellent high-resolution (.002") graphics for business and scientific applications. Cost of plotter, interface, and software is approximately $1000.
Price: $145.00
for software only
Includes line, bar, and pie chart routines (menu driven) and alphanumerics.
Available:
Strobe, Inc.
897 Independence Ave.
Bldg. 5A
Mountain View, CA 94043

Name: **Specsystem**
System: Apple II & Apple II Plus
Memory: 48K
Language: Integer BASIC or Applesoft BASIC
Hardware: Specsystem disk for use in conjunction with the Eventide A1B232 real-time audio spectrum analyzer
Description: Specsystem is for use with the Eventide A1B232 RTA and allows you to perform 3-dimensional spectral analysis and analyze reverb decay (RT-60). In addition, the new hi-res 31-band real-time analysis display yields far greater resolution and more dynamic range options than previously available. Variable period averaging and range switching are available along with other new functions.
Price: $199.00
Includes Specsystem disk and complete instruction manual.
Author: Eventide
Available:
Eventide, Inc.
265 West 54 Street
New York, NY 10019
(212) 581-9290

Name: **GL-PLUS**
System: Apple III
Memory: 128K
Language: Business BASIC
Hardware: 132-column printer and either second diskette drive or hard drive.
Description: *GL-PLUS* is an extremely flexible and easy to operate general ledger with built in receivables and payables. Reports include general ledger, month's journal, balance sheet, income statement, aged receivables and payables, receivable and payable detail, and more!
Price: $495.00
Includes operator's manual, programs, and sample company data.
Author: Dan Sargent
Available:
Great Divide Software
8060 W. Woodard Dr.
Lakewood, CO 80227

Name: **Astroids**
System: VIC 20 or TRS-80 Color Computer
Memory: 5K (VIC), 4K (TRS-80)
Language: BASIC
Hardware: Cassette
Description: This program is an arcade-style "dodge-the-asteroids" game for children 7-13 years old.
Price: $6.65 - U.S.
$7.90 - Canada
Includes program on tape, shipping.
Author: Peter and Mike James, Colin Helsby
Available:
MFJ Electro-Enterprises
P.O. Box 13076
Kanata, Ont.
K2K 1X3
(613) 592-2962 (Canada)

Name: **Snack Attack**
System: Apple II Plus
Memory: 48K
Language: Apple Machine Language
Hardware: Disk Drive
Description: An a-mazing three-maze game that's more habit forming than eating peanuts. Anyone who can play just one game deserves a medal for super self-control. You feel a Snack Attack coming on — your only hope is to gobble up as many gumdrops as you can grab away from the greedy Gumdrop Guards! Our best seller for months; one try and you're hooked. Happy Snacking!
Price: $29.95
Author: Dan Illowsky
Available:
Datamost, Inc.
9748 Cozycroft Avenue
Chatsworth, CA 91311
(213) 709-1202
or your local computer store

Name: **Hockey**
System: Atari 400/800
Memory: 16K RAM
Language: Assembler (Machine)
Hardware: Two, three, or four joysticks, cassette recorder or disk drive
Description: *Hockey* is a high-speed video action game for two, three, or four players. It is played on an enclosed rink, with scoreboard including clock overhead. Game players use joysticks to control the action. Offensive players skate with the puck, pass, and shoot. Defensive players steal the puck and intercept passes. Goalies block shots. *Hockey* includes "smart" players who perform automatically.
Price: $29.95
Available:
Gamma Software
P.O. Box 25625
Los Angeles, CA 90025
(213) 473-7441

Name: **Adventure to Atlantis**
System: Apple II
Memory: 48K
Language: Machine Code
Hardware: One disk drive
Description: This game combines the best features of adventure games, arcade games, and fantasy role-playing games. *Adventure to Atlantis* is a sequel to Odyssey: The Complete Adventure. The struggle between the forces of magic (The High One) *versus* the forces of science (The Atlanteans) continues. The game uses four methods to grab the player's attention: high-resolution color graphics and animation, sound effects to enhance the action, random events at all stages of the adventure, and embedded arcade-like action.
Price: $40.00
Includes one floppy disk.
Author: Robert C. Clardy
Available:
Synergistic Software
830 N. Riverside Drive
Suite 201
Renton, WA 98055

(Continued)

Software Catalog *(continued)*

Name: **Nightmare Gallery**
System: Apple II
Memory: 48K
Language: Machine code
Hardware: Disk Drive (paddles/joystick optional)

Description: This new fast-action game for the Apple Computer combines sound effects and high-speed arcade action into one challenging computer game. Players of Nightmare Gallery are trapped in a spooky graveyard with only a revolver to protect themselves against werewolves, ghouls, ghosts, vampire bats, and mummies. Firing a stream of silver bullets, players dodge around the graveyard. Points are awarded for hitting the supernatural terrors. A scoreboard is built into the game.

Price: $34.95
Includes one floppy disk.

Author: Ron Aldrich

Available:
Synergistic Software
830 N. Riverside Drive,
Suite 201
Renton, Washington 98055

Name: **Choplifter**
System: Apple II
Apple II Plus
Memory: 48K
Language: Assembly
Hardware: Joystick with two buttons

Description: With a mission to rescue hostages from behind enemy lines, Apple owners can now take control of a helicopter in this new arcade-style game. With realistic throttle action, players can maneuver their helicopter (in high resolution and simulated 3-D) through a sky of enemy jet fighters and air mines. Once you have successfully landed your chopper, hostages will run toward you through a barrage of tank fire and air-to-ground missiles while you frantically try to hold off the enemy. Each flight back to safety is especially nerve racking, as a full load of hostages will perish if you go down!

Price: $34.95
Includes software package.

Author: Dan Gorlin

Available:
Broderbund Software
1938 Fourth Street
San Rafael, CA 94901
(415) 456-6424
or dealers and distributors

Name: **Starship Chameleon**
System: TRS-80 Color Computer
Memory: 16K
Language: Assembly
Hardware: Cassette or disk

Description: In this graphics arcade game, you are the Starship Chameleon, a special intergalaxian vessel with the assignment of protecting the planet below from the aerial attack of enemy invaders. You have the unique capability to change color at the push of a button in order to destroy the on-coming super bombs and anti-matter bombs that have been launched by the enemy Gabolatoks above. But watch out for the semi-intelligent aerial bombs! They home in on your every move, seeking to destroy you! *Starship Chameleon* has nine levels.

Price: $24.95 - cassette
$29.95 - disk
Includes cassette or disk and instructions.

Author: Kenneth Kalish

Available:
Computerware
P.O. Box 668
4403 Manchester Ave.
Encinitas, CA 92024
(714) 436-3512

Name: **MIMIC Speech Processors**
System: Apple II Plus

Description: Low-cost speech I/O. Based on newly patented technique for digital recording and reproduction of voice signals.

Price: $199.00 for plug-in hardware.
$34.95 for MIMTALK software (prepares user vocabulary).

Available:
MIMIC, Inc.
P.O. Box 921
Acton, MA 01720

Name: **Audex**
System: Apple II (with 16K RAM Card) or Apple II Plus
Memory: 48K
Language: Applesoft
Hardware: Optional tape recorder

Description: *Audex* is a collection of utility programs that allows you to create sounds, edit them, and play them back from your own BASIC or assembly-language programs.

(Continued)

Price: $29.95
Includes instructions.

Author: Pete Kosel

Available:
Sirius Software
10364 Rockingham Drive
Sacramento, CA 95827
(916) 366-1195

Name: **Hi-Res Ultra Plotter Package MP-10**
System: Any PET/CBM Series 2001, 3000, 4000, 8000 Computer (specify)
Memory: 16K minimum
Language: BASIC Original, Upgrade, or 4.0
Hardware: Any Commodore graphics printer w/programmable line spacing

Description: This is a beautiful plotting package that takes advantage of the screen graphics and printer's ultra-high resolution graphics. Equations must be entered in BASIC. Domain and/or range may be restricted, magnified, and manipulated. Printer graphing may be very small to full page and longer having a resolution of 420 by an infinite number of dots lengthwise. Multiple equations can also be plotted. Excellent for lab or school.

Price: $12.95 cassette
$16.95 on 4040/2031 disk
Includes full instructions and step-by-step guiding through program.

Author: Louis F. Roehrs

Available:
TELE-TREX Software Systems
4 Waring Lane
Littleton, CO 80121

Name: **Firebug**
System: Apple
Memory: 48K
Language: Applesoft ROM
Hardware: Disk drive (dual DOS 3.2/3.3)

Description: Guide your mechanical Firebug through a five-level maze using your Apple keyboard or joystick, to the exit on Level 1. Your Firebug must race against time and fire to escape. A fuse is burning behind your Firebug, so your time is short! Add points for destroying the maze as you guide the Firebug through it. Fantasy effects include screen graphics of gas cans exploding in color, the sound of fire crackling, and the tic-toc of time running out.

Price: $24.95

Author: Silas S. Warner

Available:
Direct from Muse and computer stores nationwide

Name: **SpeedCOBOL and ReadCOBOL**
Memory: 64K
Language: COBOL (ANSI 74)
Hardware: Any micro, mini, or mainframe

Description: *SpeedCOBOL* is an abbreviation expander for COBOL-reserved words and phrases, data names, condition names, file names, and any other string of one to 30 characters. A direct 50% of coding and keying productivity improvements can be realized and measured with *SpeedCOBOL*. Working with abbreviations of one or two characters for COBOL words and phrases, *SpeedCOBOL* precompiles these abbreviations to their full length. *ReadCOBOL* addresses the seemingly ever-present programming resource drain of COBOL program maintenance by establishing program structure standards. These standards are maintained by *ReadCOBOL* automatically, not by the programmer. Paragraph sequencing, standard structuring and stylizing are an automatic and direct result of *ReadCOBOL*. *An optional step of renaming data names, file names, etc. is available for improved program quality. Inspecting the results of ReadCOBOL* makes the improvement in quality obvious. Paragraph sequencing, installation-wide standard programming structure and common stylizing of source statements are all direct benefits to the user.

Price: $500 for microcomputers
$5000 for mainframes
Includes documentation and diskette.

Author: Foundation for Software Engineering

Available:
Computers Et Cetera
524 W. Broadway #111
Tempe, AZ 85282
(602) 829-0888

Name: **QUNIX, "C" Compiler, Full Screen Editor**
System: Qunix
Memory: 64K
Language: "C" (Kernighan and Ritchie full implementation)
Hardware: 6809 or 8088-based microcomputers

Description: The Qunix operating system is a UNIX™-like operating system optimized for micros. Single-user, single-tasking or multi-user, multitasking versions are available together with a full "C" compiler (supports 8087 on IBM PC) and ultra fast macro full-screen editor.

Price: $625
Includes assemblers, debuggers, manuals.

Available:
Quantum Software Systems Incorporated
7219 Shea Court
San Jose, CA 95139

Name: **Audio Spectrum Analyzer**
System: TRS-80 Color Computer

Description: This program provides a colorful and graphic display of energy in sound. Audio enthusiasts can "watch the music;" color bargraphs show the relative power distribution over a nine-octave audio range. A built-in "audio kaleidoscope" mode presents random changing visual images, keyed to the tonal distribution and intensity ("color," in musical terms) of the input signal.

Price: $19.95
Includes Program Pak, instruction manual.

Available:
Radio Shack, Computer Centers and participating dealers

Name: **Computer Business Software**
System: Ohio Scientific
Language: BASIC

Description: This complete business package features order entry, inventory control, mailing labels, sales reports/history graphs, monthly cash reports, G/L distribution report, multi-state, multi-pay period payroll (with file locking), print spooling, and much more.

Price: $64.70

Available:
MicroSoftware International, Incorporated
3300 S. Madelyn
Sioux Falls, SD 57106
1-800-843-9838

Name: **Jinsam Executive**
Memory: 64K
Language: Machine
Hardware: CBM w/8050, IBM-PC

Description: This fast, powerful, flexible data-management system contains file handling, retrieval, modeling, statistical analysis, reports and labels, password entry, multi-user command files, merge, link, and join bases. It holds 65,000 records, *unlimited* fields, unlimited record lengths. Retrieves by scanning record, number, or key.

Price: $40 - User's Manual
$1295 entire unit
Entire unit includes interfaces, wordprocessing, VisiCalc, statistics, ROM, disks, pen, binder, and manual.

Author: Jim and Nancy Iscaro

Available:
Jini Micro-Systems Inc.
Box 274
Kingsbridge Station
Riverdale, NY 10463

Name: **Classified™**
System: Apple II, Apple II Plus
Memory: 48K
Language: Machine
Hardware: One disk drive minimum

Description: *Classified™* encrypts and decrypts the information stored in any standard DOS 3.3 diskette file, such as a file created by VisiCalc. The data sectors are overwritten by cipher text, leaving no trace of plain text.

Price: $39.50
Includes 24-page encryption manual.

Available:
Passage Research
945 Turquoise St.
Suite G
San Diego, CA 92109
(714) 488-5358

MICRO

Hardware Catalog

Name: **Dual-Comm Plus**
System: Apple II or Apple II Plus
Language: Apple Pascal, BASIC
Description: This two-port serial I/O card combines the features of the Apple High Speed Serial Card and the Apple Communications Card in one card with handshaking. Thumbwheel switches select the Apple slot locations. Selectable baud rates range from 50 to 19200 baud. Onboard firmware provides extensive printer and upper/lower case terminal modem support. Combined with an 80-column card, such as the FULL-VIEW 80, it turns an Apple into a full-operation terminal for a computer network system.
Price: $239.00
Includes all available features.
Available:
BIT 3 Computer Corporation
8120 Penn Ave. S.
Suite 548
Minneapolis, MN 55431
(612) 881-6955

Name: **Model 122 Industrial Trade Graphics Printer**
System: All computers
Description: Heavy-duty 120 CPS desktop graphics printer. 132 columns with pin-addressable graphics. Designed to fulfill applications such as business analysis, computer-aided designs, data processing, and timesharing computer processing.
Price: $1195 list price
Includes all standard features.
Available:
Centronics Data Computer Corporation
1 Wall St.
Hudson, NH 03051

Name: **Disk-O-Mate**
System: Commodore PET/CBM
Hardware: 2040/4040 disk drive
Description: *Disk-O-Mate* provides the user with write-protect switches/indicators for each drive, a power ON indicator, and audible error beeper. The ease of write protecting drives by switch *versus* diskette labels encourages good operating procedure to prevent accidentally erasing files. All switches and indicators are mounted in a small control box that may be placed on the disk unit or near the computer. Installation is accomplished without special tools or skills or any soldering.
Price: $70 - $80
Includes assembled unit plus installation/user instructions.
Available:
Optimized Data Systems
P.O. Box 595
Placentia, CA 92670

Name: **Olympia ESW 102**
Memory: 4K Bytes
Description: This is a high-quality, cost-effective printer developed for the letter quality printer market. Standara lettering with varying type styles, proportional spacing, four pitches, bi-directional printing, and print wheel cassette insertion. Print speed is 17.5 characters per second. Laser-sharp print quality, a choice of ribbon systems and economical pricing for end users and OEMs.
Price: $1590.00
Includes interface.
Available:
Olympia USA Inc.
Box 22
Somerville, NJ 08876

Name: **Model T Computer Slide System**
System: Apple II Plus
Memory: 48K
Language: BASIC
Hardware: Two disk drives, graphics tablet, and color monitor
Description: Product allows user to produce 35mm color slides. Sold as add-on to existing Apple II Plus with two disk drives, graphics tablet, and monitor. Product consists of a film recording device, software, and overlay for tablet. Images are created and previewed with the system. Once the desired image is created the computer transfers it to 35mm color film. The film is then removed from the film recording device and processed. The system produces business informational slides and artistic slides. Five type styles are currently available in two sizes. The color on the slides is very rich.
Price: $3495.00 (without Apple components)
Includes film recording device, software, graphics tablet overlay, interface card, and manual.
Available:
Toucan Visual Production Systems
1033 Battery Street
San Francisco, CA 94111
(415) 392-2970

Name: **Model 85 Digital Memory Oscilloscope**
System: Compatible with Apple II or Apple II Plus
Memory: 11.5K Bytes
Language: BASIC or assembly language
Description: Laboratory and automated measurements were never easier with this completely programmable digital memory oscilloscope controlled by a co-resident BASIC program. Two channels with 50 MHz bandwidth combined with full 8-bit A/D conversion provide high accuracy, high frequency measurements. Unlimited storage time, waveform storage on disk, waveform averaging, cursor DVM, hard copy output, compare to reference waveforms, and more.
Price: $995.00
Includes Model 85, user's manual, software on one 5¼" floppy disk.
Available:
Northwest Instrument Systems, Inc.
P.O. Box 1309
Beaverton, OR 97075
(800) 547-4445

Name: **I/OX-122 I/O Expansion Board**
System: 6500
Description: I/O expansion for the SYM-1, AIM-1, and other microcomputers that use 6522 VIAs for I/O and do not provide full address decoding onboard. This board has physical space for four additional 6522 VIAs, and provides additional decoding for a total of 16 devices at intervals of your choice. Connectors for all I/O lines, and further expansion are included. All 6522 functions are available, with no interference with previous functions of the original VIA. Two versions of this board are available. The I/OX-122 mounts above, and directly plugs into, an on-board 6522 socket, and relocates the original VIA to the expansion board. Where there are space limitations, the I/OX-222 uses a dip header and cable for remote installation.
Price: $49.95
OEM and dealer inquiries welcome.
Includes expansion board and one 6522.
Available:
Alternative Energy Products
P.O. Box 1019
Whittier, CA 90609

Name: **ISAAC**
System: Apple
Memory: 48K
Language: Labsoft (Extended Applesoft BASIC)
Hardware: Disk drive, monitor, printer
Description: *ISAAC* provides a hardware interface and Labsoft language to allow the Apple II Plus to interface with real world signals. Used for laboratory automation, electronic tests, or process control.
Price: $3950.00
as addition to current Apple.
$6000-$9000 for total system with Apple and various peripherals.
Available:
Cyborg Corporation
55 Chapel Street
Newton, MA 02158

(Continued)

Name: **AMDEK 13-inch "Color III" Monitor**
System: Apple II, Apple III
Hardware: Monitors, plotter, disk drive

Description: The AMDEK 13-inch color monitor provides affordable high-plus resolution graphics. It is designed with RGB video input and a commercial grade CRT; sharp and crisp color separation, 260 (H) × 300 (V) line resolution; 80- × 24-character display capability.

Price: $569.00

Available:
AMDEK Corporation
2201 Lively Blvd.
Elk Grove Village, IL 60007
(312) 364-1180
TLX: 25-4786

Name: **UP-9705 5¼" Winchester Disk Subsystem**
System: Compatible with many systems including Apple, IBM PC, Q-bus, S100, and multibus systems.

Description: This is a 5-megabyte universal 5¼" Winchester mass-storage subsystem that features interchangeable host adaptor personality cards. The unit measures 4.75 × 6.5 × 14.2 inches and weighs 16 lbs.

Price: $2,995 for one
Includes one host adaptor and documentation.

Available:
United Peripherals
432 Lakeside Drive
Sunnyvale, CA 94086
(408) 730-4440
TWX: 910-339-9359

Name: **Appli-Card**
System: Apple II
Memory: 64K on-board memory
Hardware: Z80 Microprocessor

Description: The *Appli-Card*™ is a single card that uses the CP/M operating system and can execute WordStar™ and use its full features. The hardware contributions of the *Appli-Card* are 64K of on-card memory for application development and execution. The *Appli-Card* comes standard with a 4 MHz Z80A or a 6 MHz Z80B. Also, the Z80A or Z80B CPUs can run at their maximum speed. One can have from 2K to 8K of EPROM on the card and a real time clock is supported. An expansion interface port is available for added extra memory for future versions of CP/M and to support other Z80 peripherals. The 6502 and the Z80A or Z80B processors are able to run simultaneously and at full speed. The *Appli-Card* is designed to support CP/M applications with only one card. The unique SoftVideo™ features include upper and lower case, 40-column to 255-column horizontal scrolling, and also 70 columns by 24 lines is available using the high-resolution graphics on the Apple II.

Price: $595.00
Includes *Appli-Card*, CP/M, and WordStar.

Available:
Personal Computer Products, Inc.
16776 Bernardo Center Dr.
Suite 203
San Diego, CA 92128

Name: **Ultimate RS-4/OV-1 Computer**
System: Dedicated
Memory: 64K — 48K ROM, 16K RAM
Language: Machine
Hardware: TeleVideo 912-C Terminal w/built-in board (64K) by Cannella Corp.

Description: The Ultimate computer is designed to handicap horse racing — makes a pro out of even an amateur. The *Ultimate* computes and rates according to the principle of established class, assigns each horse a specific rating, handicaps, summarizes data for up to 25 horses per race, and displays them in order of betting qualification.

Price: $6,000
$5,000 for the computer
$1,000 for Epson MX-80 Printer

Available:
Cannella Corp.
Armond F. Cannella, Pres.
420 E. Genesee St.
Suite 208
Syracuse, NY 13202

Name: **Slim 81-260**
System: Apple II
Memory: 2K RAM, 4K EPROM - expandable to 32K
Hardware: JBE Parallel Interface Card and EPROM Programmer

Description: *Slim* is a 4.5 × 6.5 single-board large-scale integration microcomputer using a 6502 microprocessor, two 6522 VIAs, four 2114 RAMs, and 2516, 2716, or 2532 EPROM. The fully-buffered 22/44-pin bus is similar to the KIM, SYM, and AIM expansion connector. Four 8-bit I/O ports connect through 16-pin DIP sockets.

Price: $199.95

Available:
John Bell Engineering
1014 Center Street
San Carlos, CA 94070

Name: **Apple Interface Breadboard**
System: Apple II
Memory: 16K Read/write, minimum configuration
Language: BASIC or assembly

Description: *The Apple Interface Breadboard* provides controlled access to those signals needed to control external devices. User can design and operate custom interfaces using BASIC. Data bus is fully buffered and protected. Up to eight decoded outputs available; up to 256 I/O devices may be addressed. A probe circuit detects logic signals and pulse edges. All interface signals available through solderless breadboard for performing experiments using hookup wire.

Price: $235.00 — Kit (BG-125K)
$310.00 — Assembled (BG-125A)
for 110V operation.
Includes power supply, all parts, BG-110 Cablecard, and text Apple Interfacing w/Exp. Required but not included — interconnect cable (BG-Cable ($14.95).

Available:
Group Technology, Ltd.
P.O. Box 87
Check, VA 24072
(703) 651-3153

Name: **212LP Direct Connect Modem**
System: RS-232

Description: The *212LP* is a 1200 bps-only full duplex, asynchronous modem. This manual answer unit requires no external AC power. All operating power is from the telephone line. It is FCC certified for direct connection to the public-switched telephone network and compatible with the high-speed channel of the Bell 212A

Price: $495.00
Includes operating manual, phone connect cable, and warranty.

Available:
Universal Data Systems
5000 Bradford Drive
Huntsville, AL 35805

Name: **PromQueen VIC Cartridge**
System: Commodore VIC-20
Memory: 4K up
Hardware: Requires expansion chassis if used with other expansions.
Language: 6502 machine code, BASIC, other uP machine codes

Description: A self contained cartridge mating to the VIC's expansion port equips the VIC-20 for 8-bit microprocessor system development. The cartridge contains 4K's of RAM accessible both to the VIC and to an external microprocessor system under development. DIP switch sets to any of 4 expansion blocks of the VIC.

Price: $169.50
Includes 24 pin textool socket and built-in 2716, 2732 and 2732A EPROM programmer with power supply. Development and EPROM utility software on 2732A EPROM, 32 page manual.

Available:
Gloucester Computer Bus Co., Inc.
6 Brooks Road
Gloucester, MA 01930
(617) 283-7719

Name: **MPC Bubble Memory Board**
System: Apple II, Apple II Plus
Memory: 1 Megabit
Language: BASIC, Pascal, CP/M

Description: *Bubble Memory Board* has 128K bytes of nonvolatile memory. It comes with software to emulate one floppy disk; autoboot PROM on board is for stand-alone pseudo disk operation.

(Continued)

Price: $895.00
Includes board/disk emulator software.
Available:
MPC Peripherals Corp.
9424 Chesapeake Drive
San Diego, CA 92123

Name: **MetaCard**
System: Apple II or Apple II Plus
Memory: 48K min.
Hardware: Disk II
Language: CP/M-86 Supplied
Description: MetaCard is a secondary processor card with: 5Mhz. Intel 8088 microprocessor, 64K bytes dynamic RAM expandable to 128K, external power supply, 2716 EPROM with power-up diagnostics and initialization logic, real-time clock, CP/M-86 operating system with optional MS-DOS or UCSD Pascal IV.0, provision for integration with future products, including 8087 piggyback board and additional 256K memory card.
Price: Introductory prices:
64K—$980, 128K—$1230 (U.S. and Canada)
Includes power supply, 64K RAM (pre-tested), CP/M-86 on Apple disks, complete documentation.
Available:
Metamorphic Systems, Inc.
P.O. Box 1541
Boulder, CO 80306

Name: **Pi-1, Pi-2, Pi-3 Video Monitors**
Description: Low-cost, high resolution monitors are available in 9" or 12" versions. Features are: 80x24 character display, resolution of 1000 lines at center, band width 20 Mhz. A front metal case provides complete shielding allowing stackability controls. It is compatible with all small business computers.
Price: Pi-1, $249; Pi-2, $275; Pi-3, $289 retail.
Available:
USI International,
Computer Products Division
71 Park Lane,
Brisbane, CA 94005
(515) 468-2900

Name: **Electric Typing Fingers (ETF-80)**
Hardware: Comes with own power supply and cables
A peripheral that turns an IBM Selectric* (or similar electronic typewriter) into a fast, letter-quality, economical printer. It is a sensible alternative to purchasing the more expensive daisy wheel printer and there are no typewriter modifications necessary.
*Selectric is a registered trademark of IBM, Incorporated.
Price: $495.00
Includes power supply and connecting cable.
Available:
Personal Micro Computers, Inc.
475 Ellis Street
Mt. View, CA 94043
(415) 962-0220

Name: **PKASO ID12 -Color Interface**
System: Apple II or Apple III Computers
Hardware: ID12 - Color Interface
Description: The ID12 - color Interface converts your Apple Computer and IDS Color Prism printer to a complete text and graphic output system. Features include: full snapshot dump of any screen image, hi-res and lo-res screen prints using accurate screen colors, quick commands to swap and rearrange the color set, quick commands to change printing colors even within the word processor text.
Price: $195.00 - $225.00
Includes cable, instructional diskette and comprehensive manual.
Available:
Interactive Structures
146 Montgomery Ave.
P.O. Box 404
Bala Cynwyd, PA 19004
(215) 667-1713

Name: **Model 140-RS Computer Interface**
System: Accepts serial data from computer to operate random access slide projector, Kodak or Mast
Description: This interface is designed to convert an asynchronous-serial data stream into signals usable by Master Kodak random-access slide projectors.

(Continued)

Hardware Catalog *(continued)*

Price: $621.00
Includes interface package, connectings cable.
Available:
Mast Development Co.
2212 East 12th Street
Davenport, IO 52803

Name: **Corvus Systems Disks**
System: Atari 800
Memory: 6 Megabits
Hardware: Winchester Disk
Description: The Corvus System 6 MB hard-disk drive interfaces with Atari 800. It has two plotters and four data surfaces; the diameter is 5¼ inches.
Price: $3,195
Includes Z80 microprocessor, controller, managing bidirectional data transfer.
Available:
Corvus Systems
2029 O'Toole Ave.
San Jose, CA 95131
(408) 946-7700

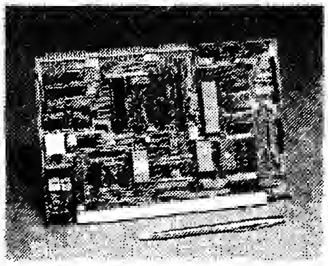
Name: **Slot 8**
System: Apple II, Apple II Plus
Memory: 16K
Language: BASIC, Pascal, CP/M, Assembly (6809, 6502, etc.)
Description: *Slot 8* is a space-saver for the Apple II and provides the user with an extra slot for peripheral use. *Slot 8* plugs into slot 7 and provides two slots on the Apple motherboard. Most peripheral cards for the Apple will work in *Slot 8.*
Includes *Slot 8* card and manual of operation.
Available:
Legend Industries, Ltd.
2220 Scott Lake Rd.
Pontiac, MI 48054
(313) 674-0954

Name: **DATASAVER™ AC Power Backup Unit**
System: Any
Description: *DATASAVER™* provides uninterrupted AC power to microcomputer systems and instruments. The 90 watt capacity unit is for Apple II/III and TRS-80 III systems. The 200 watt unit covers larger systems including 5¼" fixed disk drives. The desk-top unit is just 4" × 6" × 9½" and has internal battery for five minutes of automatic power hold-up.
Price: $395 - 90 watt
$695 - 200 watt
Includes overvoltage protection and power status indicator LED, buzzer, and interrupt signal.
Available:
Cuesta Systems, Inc.
3440 Roberto Court
San Luis Obispo, CA 93401
(805) 541-4160

Name: **2809 Softboard System**
System: 6809 SS50 Bus computers (SWTPC, Gimix, Smoke Signal)
Description: The 2809 softboard system enables 6809 SS50 Bus users to immediately run over 2000 CP/M application programs on their computers
Price: $595.00
Includes 50 pin processor board, CP/M 2.2, editor, assembler, debugger, and complete reference manuals.
Available:
Meta Lab
P.O. Box 1559
Suite 106
2888 Bluff Street
Boulder, CO 80301
(303) 499-4236

Name: **M 78-5 TrimForm Printer**
Memory: Line buffer, or 1K data buffer
Language: ASCII
Description: This printer has 140 characters per second, tractor feed for forms up to 9.5" wide, 96-character set (upper/lower), OCR-A (optional), bar code, vertical forms control unit, quiet 56 dbA, RS-232 serial interface, integral form cutting device under host control, and self test.
Price: $4345
Includes all hardware.
Available:
Mannesmann Tally
8301 South 180th St.
Kent, WA 98031
(206) 251-5524

MICRO



How to run a listing in Software/Hardware

The Software and Hardware Catalogs are provided as a service both to our readers and to the manufacturers. These entries are not MICRO reviews, but descriptions provided by the manufacturer.

To run a free listing in either catalog, a company fills out the appropriate form or merely mails in their material in the same format that appears in the magazine.

We try to limit entries to one company per month, on a first-come-first-serve basis.

If you sell products our readers should know about, write to Software/Hardware Catalog, MICRO, P.O. Box 6502, Chelmsford, MA 01824.

READER SURVEY: SEPTEMBER 1982

WE NEED YOUR HELP! To keep MICRO in touch with the rapidly changing computer world so that we can give you the information you need, please take a minute to fill in this questionnaire and mail it back to us. THANK YOU for your time.

1. How long have you subscribed to or read MICRO?
□ Less than 6 months □ 6 months to 1 year □ Over 1 year □ Over 2 years □ Over 3 years □ From the beginning

2. How did you get your current issue?
□ Subscription □ Computer store □ Newsstand □ Bookstore □ Borrowed □ Library

DEMOGRAPHICS

3. What is your age?
□ – 17 □ 18-25 □ 26-35 □ 36-45 □ 46-55 □ 56-65 □ 65 +

4. What is your occupation?
□ Programmer/analyst □ Professor/teacher □ Business person
□ Engineer □ Lawyer □ Student
□ Technician □ Doctor □ Other ____________

5. What is your formal educational level?
□ Fewer than 12 years □ High school graduate □ Associate degree □ Bachelor's degree □ Para-professional degree
□ Advanced degree

6. What is your annual household income before taxes?
□ Less than $15,000 □ $15,000-19,999 □ $20,000-24,999 □ $25,000-34,999 □ $35,000-44,999 □ $45,000-54,999 □ $55,000 +

COMPUTER INFORMATION

7. What microcomputer(s) do you use?
□ AIM □ Apple II □ Apple III □ Atari (please specify model) ________ □ KIM □ OSI (please specify model) ____________
□ CBM/PET □ SuperPET □ VIC □ SYM □ TRS-80 Color Computer □ Other 6502 □ Other 6809 ____________
□ Other processor(s) ____________

8. Where do you use the above computer(s)?
□ Home □ Work □ School □ Other ____________

9. Approximately how much have you spent on your computer equipment so far?
□ – $500 □ $500-999 □ $1,000-1,999 □ $2,000-2,999 □ $3,000-3,999 □ $4,000-4,999 □ $5,000-9,999 □ $10,000 +

10. Approximately how much do you expect to spend on your computer equipment in the next year?
□ – $500 □ $500-999 □ $1,000-1,999 □ $2,000-2,999 □ $3,000-3,999 □ $4,000-4,999 □ $5,000-9,999 □ $10,000 +

11. Approximately how much have you spent on your computer software so far?
□ – $200 □ $200-499 □ $500-999 □ $1,000-1,999 □ $2,000 +

12. Approximately how much do you expect to spend on computer software in the next year?
□ – $200 □ $200-499 □ $500-999 □ $1,000-1,999 □ $2,000 +

13. What sorts of additions to your basic system have you made?
□ Disk Drives □ Modem □ Serial Interface □ Parallel Interface □ RAM cards □ 6809 card □ 68000 card □ Z80 card
□ Printer (type) ____________ □ Other ____________

14. What additional changes or upgrades would you like to add to your system?
□ Disk Drives □ Modem □ Serial Interface □ Parallel Interface □ RAM cards □ 6809 card □ 68000 card □ Z80 card
□ Printer (type) ____________ □ Other ____________

15. How do you use your computer equipment?
□ Business □ Software Development □ Hardware Development □ Telecommunications □ Entertainment □ Hobby □ Education
□ Communications □ Word Processing (type) ____________ □ Database Management ____________

16. What languages do you use?
□ BASIC □ Pascal □ FORTH □ COBOL □ APL □ LOGO □ LISP □ 6502 Assembler □ 6809 Assembler
□ 68000 Assembler □ Other ____________

17. In an average week, about how many hours do you spend on a microcomputer performing the following operations?

	0-2	2-4	4-8	8-10	More
Programming for fun or self-education	□	□	□	□	□
Programming professionally	□	□	□	□	□
Using packaged programs in business	□	□	□	□	□
Using packaged programs at home	□	□	□	□	□
Using packaged programs for education	□	□	□	□	□
Playing games	□	□	□	□	□
Other ____________	□	□	□	□	□

18. If you write programs, what type of programming do you spend most of your time developing?
□ Business applications □ Games □ Software development utilities

19. In an average month how much time do you spend with MICRO?
□ Less than 2 hours □ 2-4 hours □ 4-8 hours □ More than 8 hours

20. How would you rate your present microcomputer knowledge?
Software: □ Elementary □ Intermediate □ Advanced
Hardware: □ Elementary □ Intermediate □ Advanced

21. Have you ever constructed a computer or computer equipment? □ Yes □ No
If yes, describe ____________
____

22. Have you switched from one computer to another? □ Yes □ No
If yes, explain ____________

(Continued on reverse side)

MAGAZINE

23. To what other computer publications do you subscribe?
□ BYTE □ Compute! □ □Dr. Dobbs □ Kilobaud Microcomputing □ Nibble □ Softside □ Softalk
□ Popular Computing □ Personal Computing □ 80 Microcomputing □ 68 Micro □ Interface Age □ Transactor
□ Other(s) ____________________

24. Please rate the following parts of MICRO as to their interest, with 5 being very interesting and 1 not at all interesting.

	5	4	3	2	1		5	4	3	2	1		5	4	3	2	1
News	□	□	□	□	□	Hardware & Software Catalogs	□	□	□	□	□	Advertisements	□	□	□	□	□
Articles	□	□	□	□	□	System specific information	□	□	□	□	□	Columns	□	□	□	□	□
Reviews	□	□	□	□	□	New Publications	□	□	□	□	□	Editorials	□	□	□	□	□

25. Please rate the following kinds of articles as to their interest, with 5 being very interesting and 1 not at all interesting.

	5	4	3	2	1		5	4	3	2	1		5	4	3	2	1
Hardware tutorials	□	□	□	□	□	Applications	□	□	□	□	□	Review articles	□	□	□	□	□
Programming techniques	□	□	□	□	□	Games	□	□	□	□	□						
Utilities	□	□	□	□	□	News & Information	□	□	□	□	□						

26. Is MICRO □ too technical □ not technical enough □ just right?

27. Have you written articles for publication? □ Yes □ No

28. Would you be willing to pay extra to receive MICRO's programs in another form? □ Yes □No

29. If yes to #28, which form would you prefer?
□ Printed □ Disk □ The Source □ MicroNet □ Other ____________________

30. Overall, how do you feel about MICRO? How useful is MICRO to you?
____
____
____
____

Fold Here and Staple Closed

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 60, CHELMSFORD, MA

POSTAGE WILL BE PAID BY ADDRESSEE

MICRO
34 Chelmsford Street
P.O. Box 6502
Chelmsford, MA 01824

FOCUS ON THE 6809 MICRO WORTH SWITCHING FOR

FOCUS ... ON ITS SUPERIOR HARDWARE

- 6809E Microprocessor provides 8-bit economy with 16-bit capabilities, position independent code and many advanced features
- Commercial quality Keyboard with full UPPER/lower case ASCII, numeric keypad and cursor control keys
- Two double-sided double-density mini diskettes with 640K bytes of IBM-compatible storage
- 56K Bytes User RAM
- Full Video with programmable screen formats, bit mapped graphics, user definable character sets, reverse video and hardware scrolling, plus a light pen interface
- Full Communications support for RS-232 at 50 to 19.2K baud with programmable data formats
- Built-in I/O Drivers include 6 parallel I/O ports, 3 serial I/O ports, 6 timers/counters, 20 mA current loop, and a programmable cassette interface

FOCUS ... ON ITS EXTRAORDINARY SOFTWARE

- Choice of two 'Universal' 6809 Disk Operating Systems supported by many companies: FLEX ™ or OS-9™
- Extended Floating Point Disk BASIC Interpreted and Compiled
- Text Editor and Macro Assembler
- Full feature FlexiMon debugging monitor
- Built-in Word Processor

FOCUS ... ON ITS VERSATILE EXPANDABILITY

- IEEE-488 Bus Controller option for instrumentation
- Supports multiple 8" diskette drives
- PASCAL, FORTH and other high level programming languages are available
- Complete Application Packages are available from many software sources
- Add-on Hardware includes: Date/Time, Opto-Isolators, Stepper-Motor Controller, A/D and D/A, RAM and EPROM Memory, and others from several manufacturers

FOCUS ... ON YOUR APPLICATION REQUIREMENTS

- FOCUS provides a completely integrated system of hardware and software, so that you can concentrate on your application requirements

*Licensing arrangement for OS-9 pending.
Please contact The COMPUTERIST, Inc. for further details.

THIS IS FOCUS™

$\$3495^{00}$

VERY COMPLETE

THE COMPUTERIST®

34 Chelmsford Street
Chelmsford, MA 01824
Phone: 617/256-3649
Telex: 955318 INTL DIV

DR. DOBB'S JOURNAL

For Users of Small Computer Systems

with topics like:

Small-C
FORTH
Lisp
CP/M
S-100
Cross-assemblers
8086/8088
16-bit tech.
Pascal
Assembler
Compilers
Algorithms
8080/Z80
6502
etc.

Each issue includes:

- **valuable software tools**
- **algorithms & problem solving**
- **industry news**
- **important product reviews**

With in depth coverage of:

- **telecommunications**
- **systems programming**
- **language development**
- **machine independent programs**

and much, much more!

Yes! Please enter my subscription for T4
□ 1 yr. (12 issues) $25 □ 2 yrs. $47 (save $13 off newsstand)
□ Please bill me □ I enclose check/money order

Name ______________________________

Address ______________________________

City ______________ State ______ Zip ______

Are you ready?

DDJ, the world's foremost microcomputer publication, has been working for years to prepare its readers to be innovators, to lead the wave of breakthroughs in our changing technology.

Every issue of *Dr. Dobb's Journal* helps one to understand the nuts and bolts of small computer systems. We offer entire listings of valuable software: our pages have included compilers,* cross-assemblers, editors, new languages, hardware interfaces and more – usually before anyone else thinks of them!

Even more important!

As valuable and significant as all those things are, there is an even grea er reason for you to join forces with *DDJ*. That is the keen, responsive readership. Our subscribers share insights, correspond, and contribute tc one another's work, more than any other group we know. They treat *Dr. Dobb's Journal* as a "hands-on" publication.

This warm cooperation has done more to refine software products, an generally to advance the state of microcomputer technology, than per haps any other resource. And it is available to you through our pages!

For the straight Facts . . .

If you are a serious computing professional or enthusiast, then you should take a very close look at what *DDJ* offers you. We've been on the cutting edge since 1976.

Dr. Dobb's Journal • A Publication of People's Computer Company, 1263 El Camino Real, P.O. Box E, Menlo Park, CA 94025

MC68000

MC68000 16-bit Microprocessor

Manufactured by Motorola. 16-bit design increases speed and addressing capability. Well-suited to high-level operating systems, including UNIX, and to modular, reentrant, and relocatable programming.

Design based on PDP-11 minicomputer, includes:

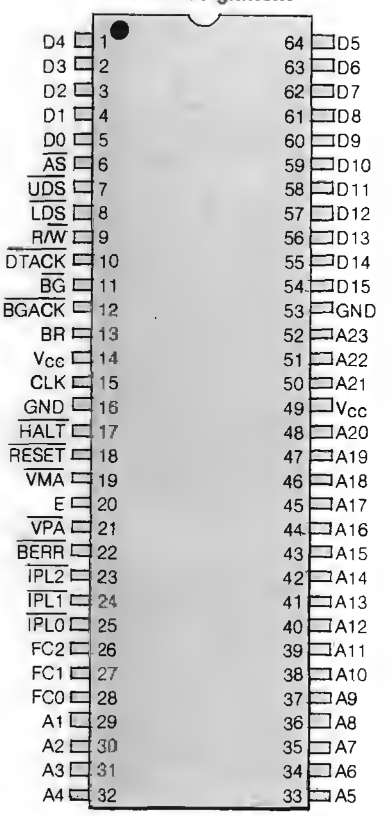
32-bit data and address registers
16 megabyte direct addressing range
56 different instructions
14 different addressing modes for most instructions
5 main data types: bit, byte, word, long word, and decimal
Memory-mapped I/O

Programming model includes two 32-bit stack pointers, 32-bit program counter, 8-bit status register, eight 32-bit data registers, and seven 32-bit address registers. The eight data registers, seven address registers, and two stack pointers may all be used as index registers.

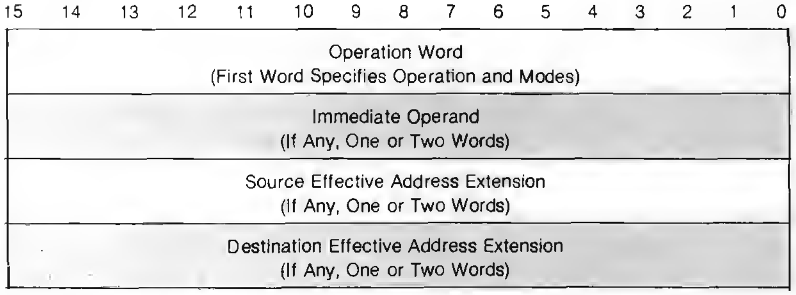
Included in many new computers, including TRS-80 Model 16, Corvus Concept, SAGE II, and Fortune 32:16.

Included in add-on boards manufactured by DTACK Grounded, Motorola, and others for Apple, PET, and other 8-bit computers.

Pin Assignment



Instruction Format



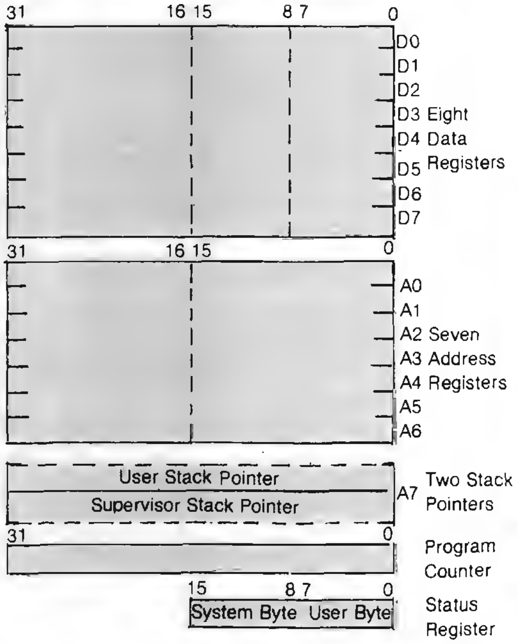
Operation Word
(First Word Specifies Operation and Modes)

Immediate Operand
(If Any, One or Two Words)

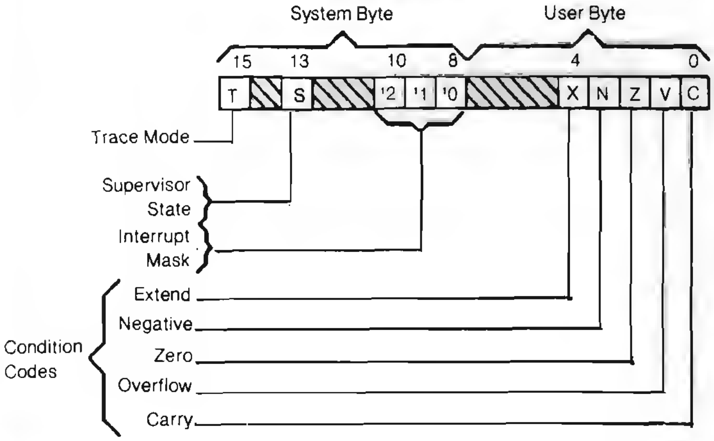
Source Effective Address Extension
(If Any, One or Two Words)

Destination Effective Address Extension
(If Any, One or Two Words)

Programming Model



Status Register



MC68000

Instruction Set

Mnemonic	Description
ABCD	Add Decimal with Extend
ADD	Add
AND	Logical And
ASL	Arithmetic Shift Left
ASR	Arithmetic Shift Right
Bcc	Branch Conditional
BCHG	Bit Test and Change
BCLR	Bit Test and Clear
BRA	Branch Always
BSET	Bit Test and Set
BSR	Branch to Subroutine
BTST	Bit Test
CHK	Check Register Againat Bounds
CLR	Clear Operand
CMP	Compare
DBcc	Test Condition, Decrement and Branch
DIVS	Signed Divide
DIVU	Unsigned Divide

Mnemonic	Description
EOR	Exclusive OR
EXG	Exchange Registers
EXT	Sign Extend
JMP	Jump
JSR	Jump to Subroutine
LEA	Load Effective Address
LINK	Link Stack
LSL	Logical Shift Left
LSR	Logical Shift Right
MOVE	Move
MOVEM	Move Multiple Registers
MOVEP	Move Peripheral Data
MULS	Signed Multiply
MULU	Unsigned Multiply
NBCD	Negate Decimal with Extend
NEG	Negate
NOP	No Operation
NO	Ones Complement

Mnemonic	Description
OR	Logical OR
PEA	Push Effective Address
RESET	Reset External Devices
ROL	Rotate Left without Extend
ROR	Rotate Right without Extend
ROXL	Rotate Left with Extend
ROXR	Rotate Right with Extend
RTE	Return from Exception
RTR	Return and Restore
RTS	Return from Subroutine
SBCD	Subtract Decimal with Extend
Scc	Set Conditonal
STOP	Stop
SUB	Subtract
SWAP	Swap Data Register Halves
TAS	Test and Set Operand
TRAP	Trap
TRAPV	Trap on Overflow
TST	Test
UNLK	Unlink

Variations of Instruction Types*

Instruction Type	Variation	Description
ADD	ADD	Add
	ADDA	Add Address
	ADDQ	Add Quick
	ADDI	Add Immediate
	ADDX	Add with Extend
AND	AND	Logical AND
	ANDI	AND Immediate
	ANDI to CCR	AND Immediate to Condition Code
	ANDI to SR	AND Immediate to Status Register
CMP	CMP	Compare
	CMPA	Compare Address
	CMPM	Compare Memory
	CMPI	Compare Immediate
EOR	EOR	Exclusive OR
	EORI	Exclusive OR Immediate
	EORI to CCR	Exclusive Immediate to Condition Codes
	EORI to SR	Exclusive OR Immediate to Status Register

*Additional instructions — actually special variations of those above.

Instruction Type	Variation	Description
MOVE	MOVE	Move
	MOVEA	Move Address
	MOVEO	Move Ouick
	MOVE to CCR	Move to Condition Codes
	MOVE to SR	Move to Status Register
	MOVE from SR	Move from Status Register
	MOVE to USP	Move to User Stack Pointer
NEG	NEG	Negate
	NEGX	Negate with Extend
OR	OR	Logical OR
	ORI	OR Immediate
	ORI to CCR	OR Immediate to Condition Codes
	ORI to SR	OR Immediate to Status Register
SUB	SUB	Subtract
	SUBA	Subtract Address
	SUBI	Subtract Immediate
	SUBO	Subtract Ouick
	SUBX	Subtract with Extend

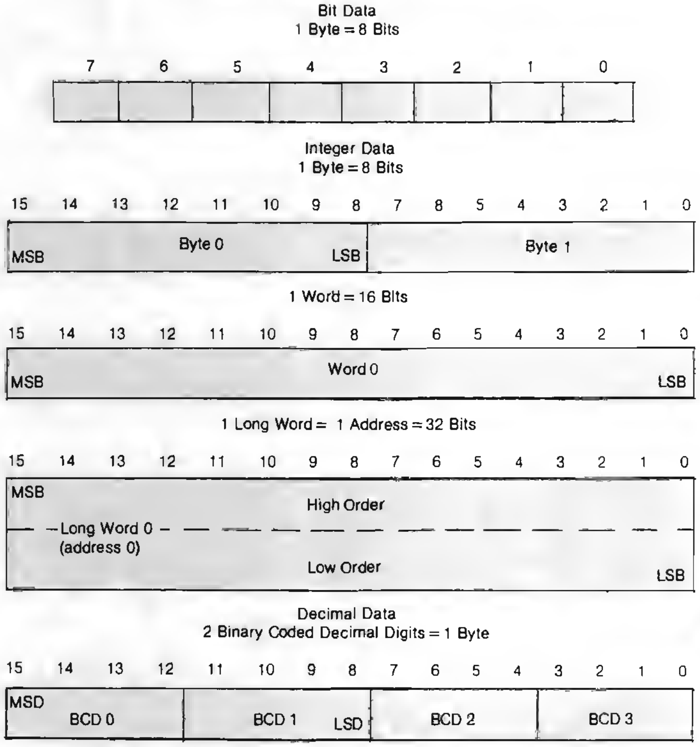
Data Addressing Modes

Mode	Generation
Register Direct Addressing	
Data Register Direct	EA = Dn
Address Register Direct	EA = An
Absolute Data Addressing	
Absolute Short	EA = (Next Word)
Absolute Long	EA = (Next Two Words)
Program Counter Relative Addressing	
Relative with Offset	EA = (PC) + d_{16}
Relative with Index and Offset	EA = (PC) + (Xn) + d_8
Register Indirect Addressing	
Register Indirect	EA = (An)
Postincrement Register Indirect	EA = (An), An ← An + N
Predecrement Register Indirect	An ← An − N, EA = (An)
Register Indirect with Offset	EA = (An) + d_{16}
Indexed Register Indirect with Offset	EA = (An) + (Xn) + d_8
Immediate Data Addressing	
Immediate	DATA = Next Word(s)
Ouick Immediate	Inherent Data
Implied Addressing	
Implied Register	EA = SR, USP, SP, PC

Notes:

EA	= Effective Address	PC	= Program Counter
An	= Address Register	( )	= Contents of
Dn	= Data Register	d_8	= 8-bit Offset (displacement)
Xn	= Address or Data Register used as Index Register	d_{16}	= 16-bit Offset (displacement)
SR	= Status Register	N	= 1 for Byte, 2 for Words and 4 for Long Words
		←	= Replaces

Data Organization In Memory



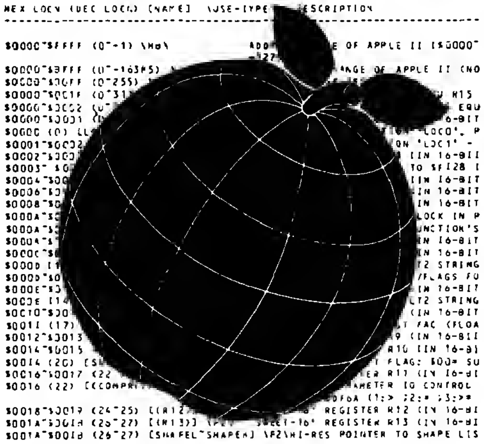
Announcing ....A New, Authoritative Guide to The Most Important Book Ever Published for the Apple.

What's Where in the APPLE...Plus...the All New Guide to What's Where

is William F. Luebbert's Revised Edition of the famous Apple Atlas. The original *What's Where in the APPLE?* provided more information on the Apple's memory than was available anywhere else. Now the *Revised Edition* shows you how to use this valuable data.

What's Where in the APPLE...Plus... the All New Guide to What's Where

- Guides you — with a numerical Atlas and an alphabetical Gazetteer — to over 2,000 memory locations of PEEKs, POKEs, and CALLs.
- Gives names and locations of various Monitor, DOS, Integer BASIC, and Applesoft routines — and tells you what they're used for.
- Explains how effectively to use the information contained in the original *What's Where in the Apple?*
- Enables you to move easily between BASIC and Machine Language.
- Guides you through the inner workings and hidden mechanisms of the Apple.



All Apple users will find this book helpful in understanding their machine, and essential for mastering it!

The Atlas and The ALL NEW GUIDE are available in one, 256-page Wire-O-Bound book for only **$24.95**

If you own the original *What's Where in the Apple?* you will want **THE GUIDE** to complement your edition. This 128-page, Wire-O-Bound version contains all new material to be used with the memory map and atlas for $9.95

Ask for it at your computer store

Use the Coupon to Order Direct from MICRO
or
Call Toll Free Today 1-800-345-8112
(In PA 1-800-662-2444)

Please send me: M982

_____ What's Where in the APPLE...Plus... the All New Guide to What's Where @ $24.95

_____ THE GUIDE @ $ 9.95

Add $2.00 surface shipping for each copy,
Massachusetts residents add 5% sales tax.

Total Enclosed $ ________

□ Check □ VISA □ Master Card Acct # ________

Expires ________

____
Name

____
Address

____
City State Zip

MICRO INK 34 Chelmsford St., P.O. Box 6502, Chelmsford, MA 01824

83-345

AARDVARK – THE ADVENTURE PLACE

ADVENTURES FOR OSI, TRS-80, TRS-80 COLOR, SINCLAIR, PET, VIC-20

ADVENTURES – Adventures are a unique form of computer game. They let you spend 30 to 70 hours exploring and conquering a world you have never seen before. There is little or no luck in Adventuring. The rewards are for creative thinking, courage, and wise gambling – not fast reflexes.

In Adventuring, the computer speaks and listens to plain English. No prior knowledge of computers, special controls, or games is required so everyone enjoys them—even people who do not like computers.

Except for Quest, itself unique among Adventure games, Adventures are non-graphic. Adventures are more like a novel than a comic book or arcade game. It is like reading a particular exciting book where you are the main character.

All of the Adventures in this ad are in Basic. They are full featured, fully plotted adventures that will take a minimum of thirty hours (in several sittings) to play.

Adventuring requires 16k on Sinclair, TRS-80, and TRS-80 Color. They require 8k on OSI and 13k on VIC-20. Sinclair requires extended BASIC.

TREK ADVENTURE by Bob Retelle – This one takes place aboard a familiar starship and is a must for trekkies. The problem is a familiar one – The ship is in a "decaying orbit" (the Captain never could learn to park!) and the engines are out (You would think that in all those years, they would have learned to build some that didn't die once a week). Your options are to start the engine, save the ship, get off the ship, or die. Good Luck.

Authors note to players – I wrote this one with a concordance in hand. It is very accurate – and a lot of fun. It was nice to wander around the ship instead of watching it on T.V.

CIRCLE WORLD by Bob Anderson – The Alien culture has built a huge world in the shape of a ring circling their sun. They left behind some strange creatures and a lot of advanced technology. Unfortunately, the world is headed for destruction and it is your job to save it before it plunges into the sun!

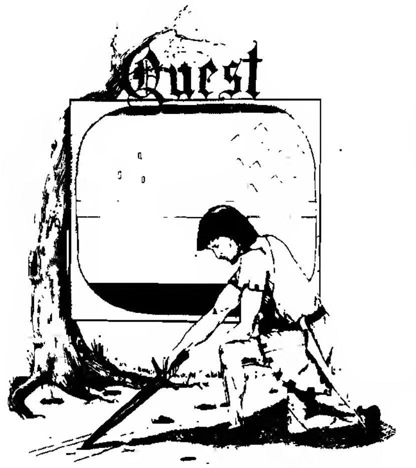
Editors note to players – In keeping with the large scale of Circle World, the author wrote a very large adventure. It has a lot of rooms and a lot of objects in them. It is a very convoluted, very complex adventure. One of our largest. Not available on OSI.

HAUNTED HOUSE by Bob Anderson – This one is for the kids. The house has ghosts, goblins, vampires and treasures – and problems designed for the 8 to 13 year old. This is a real adventure and does require some thinking and problem solving – but only for kids.

Authors note to players—This one was fun to write. The vocabulary and characters were designed for younger players and lots of things happen when they give the computer commands. This one teaches logical thought, mapping skills, and creativity while keeping their interest.

DERELICT by Rodger Olsen and Bob Anderson – For Wealth and Glory, you have to ransack a thousand year old space ship. You'll have to learn to speak their language and operate the machinery they left behind. The hardest problem of all is to live through it.

Authors note to players – This adventure is the new winner in the "Toughest Adventure at Aardvark Sweepstakes". Our most difficult problem in writing the adventure was to keep it logical and realistic. There are no irrational traps and sudden senseless deaths in Derelict. This ship was designed to be perfectly safe for its' builders. It just happens to be deadly to alien invaders like you.



NUCLEAR SUB by Bob Retelle – You start at the bottom of the ocean in a wrecked Nuclear Sub. There is literally no way to go but up. Save the ship, raise her, or get out of her before she blows or start WWIII.

Editors note to players—This was actually plotted by Rodger Olsen, Bob Retelle, and someone you don't know – Three of the nastiest minds in adventure writing. It is devious, wicked, and kills you often. The TRS-80 Color version has nice sound and special effects.

EARTHQUAKE by Bob Anderson and Rodger Olsen – A second kids adventure. You are trapped in a shopping center during an earthquake. There is a way out, but you need help. To save yourself, you have to be a hero and save others first.

Authors note to players – This one feels good. Not only is it designed for the younger set (see note on Haunted House), but it also plays nicely. Instead of killing, you have to save lives to win this one. The player must help others first if he/she is to survive – I like that.

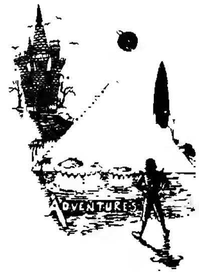
PYRAMID by Rodger Olsen – This is one of our toughest Adventures. Average time through the Pyramid is 50 to 70 hours. The old boys who built this Pyramid did not mean for it to be ransacked by people like you.

Authors note to players – This is a very entertaining and very tough adventure. I left clues everywhere but came up with some ingenous problems. This one has captivated people so much that I get calls daily from as far away as New Zealand and France from bleary eyed people who are stuck in the Pyramid and desperate for more clues.

QUEST by Bob Retelle and Rodger Olsen – THIS IS DIFFERENT FROM ALL THE OTHER GAMES OF ADVENTURE!!!! It is played on a computer generated map of Alesia. You lead a small band of adventurers on a mission to conquer the Citadel of Moorlock. You have to build an army and then arm and feed them by combat, bargaining, exploration of ruins and temples, and outright banditry. The game takes 2 to 5 hours to play and is different each time. The TRS-80 Color version has nice visual effects and sound. Not available on OSI. This is the most popular game we have ever published.

MARS by Rodger Olsen – Your ship crashed on the Red Planet and you have to get home. You will have to explore a Martian city, repair your ship and deal with possibly hostile aliens to get home again.

Authors note to players – This is highly recommended as a first adventure. It is in no way simple—playing time normally runs from 30 to 50 hours – but it is constructed in a more "open" manner to let you try out adventuring and get used to the game before you hit the really tough problems.



ADVENTURE WRITING/DEATHSHIP by Rodger Olsen – This is a data sheet showing how we do it. It is about 14 pages of detailed instructions how to write your own adventures. It contains the entire text of Deathship. Data sheet - $3.95. NOTE: Owners of OSI, TRS-80, TRS-80 Color, and Vic 20 computers can also get Deathship on tape for an additional $5.00.

PRICE AND AVAILABILITY:

All adventures are $14.95 on tape except Earthquake and Haunted House which are $9.95. Disk versions are available on OSI and TRS-80 Color for $2.00 additional.

Please specify system on all orders

ALSO FROM AARDVARK – This is only a partial list of what we carry. We have a lot of other games (particularly for the TRS-80 Color and OSI), business programs, blank tapes and disks and hardware. Send $1.00 for our complete catalog.

AARDVARK - 80
2352 S. Commerce, Walled Lake, MI 48088
(313) 669-3110

Phone Orders Accepted 8:00 a.m. to 4:00 p.m. EST. Mon.-Fri.

TRS-80 COLOR SINCLAIR OSI VIC-20

NATIONAL ADVERTISING REPRESENTATIVES

WEST COAST

The R.W. Walker Co., Inc.
2716 Ocean Park Boulevard
Suite 1010
Santa Monica, California 90405
(213) 450-9001

serving: Washington, Oregon, Idaho, Montana, Wyoming, Colorado, New Mexico, Arizona, Utah, Nevada, California, Alaska, and Hawaii (also British Columbia and Alberta, Canada).

MID-WEST TERRITORY

Cem Com Associates
Chuck Moodhe
65 E. Palatine Road
Suite 321
Prospect Heights, Illinois 60070
(312) 459-7311

serving: North Dakota, South Dakota, Nebraska, Kansas, Missouri, Indiana, Illinois, Iowa, Michigan, Wisconsin, and Minnesota.

MIDDLE ATLANTIC AND SOUTHEASTERN STATES

Dick Busch Inc.

Richard V. Busch
6 Douglass Dr., R.D. #4
Princeton, NJ 08540
(201) 329-2424

Eleanor M. Angone
74 Brookline
E. Atlantic Beach, NY 11561
(516) 432-1955

serving: New York, Pennsylvania, New Jersey, Delaware, Maryland, West Virginia, Virginia, D.C., North Carolina, South Carolina, Louisianna, Tennessee, Mississippi, Alabama, Georgia, and Florida.

NEW ENGLAND AND ALL OTHER TERRITORIES

Kevin B. Rushalko
Peterboro, New Hampshire 03458
(603) 547-2970

serving: Maine, New Hampshire, Vermont, Massachusetts, Rhode Island, Connecticut, Ohio, Kentucky, Oklahoma, Arkansas, and Texas (also any other territories not listed above).

ADVERTISING MANAGER

Cathi Bland

address materials directly to:
MICRO INK, Advertising
34 Chelmsford Street
Chelmsford, Massachusetts 01824
(617) 256-5515

Advertiser's Index

MICRO INK is not responsible for claims made by its advertisers. Any complaint should be submitted directly to the advertiser. Please also send written notification to MICRO.

Next Month in MICRO

October: Education Feature

- **Three Faces of Apple LOGO —** An in-depth discussion of the three available versions of the educational computer, LOGO.
- **A Personal Computer System for Untrained Users —** A look at Applesoft BASIC programs that assist untrained persons in using an unsupervised computer.
- **One Teacher's PET —** The author, a fifth grade teacher, discusses the growing involvement of PET computers in her classroom.
- **Introduction to Turtle Graphics —** This article describes the Turtle Graphics routines available with Apple Pascal. A graphics screen dump is also included.

Department Highlights

Data Sheet
Apple Slices
PET Vet
New Publications
Reviews in Brief
Software and Hardware Catalogs

More articles...

Formatted Output on the Atari
Apple Pascal P-Code and the 6809
PET Machine-Language Screen Utilities
The IBM Selectric as a Printer for the C1P

The
That's an Open Door to the Computer Future

KIDS & THE APPLE is its name, and its game is to prepare your child, or any child, to take his or her place as a member of the computer generation by teaching them the mysteries of the Apple* computer in ways they'll love and enjoy. Don't be surprised if you will also learn along with your child.

The kids of today are fascinated by computers to start with. And that's great, because it means **they're eager** to learn. But, until this book by Edward H. Carlson, learning about the Apple was a fumbling, bumbling effort for a child.

KIDS & THE APPLE was designed in every aspect to lead them gently, interestingly yet quickly into the computer world. First, it's a large 8½ by 11 book which can be opened flat for ease of use. Second, there are 35 chapters, each one building upon the knowledge of the prior chapter – and it's loaded with dozens and dozens of cartoons which make a point as they amuse.

At computer stores, or from:
The educational/book division of
DATAMOST INC
9748 Cozycroft Ave.
Chatsworth, CA 91311 (213) 709-1202

Then, there are special sections for a parent or teacher to use so they can work along with the kids, if they wish, and help them over any rough spots.

Perhaps the major reasons the kids will love this book is that it is **truly** written so they can easily understand it (without a lot of confusing technical language) . . . and that they see on-screen-results almost immediately! Right away they realize they'll soon be programming their Apple, making their own games! . . . or creating other programs for school or work or to play.

The computer world is roaring toward us. To be successful at work, school or even play, a child will have to be knowledgeable about computers. Make sure your favorite child is prepared for the challenge. With KIDS & THE APPLE at his side, he'll enjoy learning and you'll know you've prepared him or her for a successful future. Only $19.95.

*Apple is a trademark of Apple Computer, Inc.

THE PROWRITER COMETH.

(And It Cometh On Like Gangbusters.)

Evolution.

It's inevitable. An eternal verity.

Just when you think you've got it knocked, and you're resting on your laurels, somebody comes along and makes a dinosaur out of you.

Witness what happened to the Centronics printer when the Epson MX-80 came along in 1981.

And now, witness what's happening to the MX-80 as the ProWriter cometh to be the foremost printer of the decade.

SPEED

MX-80: 80 cps, for 46 full lines per minute throughput.
PROWRITER: 120 cps, for 63 full lines per minute throughput.

GRAPHICS

MX-80: Block graphics standard, fine for things like bar graphs.
PROWRITER: High-resolution graphics features, fine for bar graphs, smooth curves, thin lines, intricate details, etc.

PRINTING

MX-80: Dot matrix business quality.
PROWRITER: Dot matrix correspondence quality, with incremental printing capability standard.

FEED

MX-80: Tractor feed standard; optional friction-feed kit for about $75 extra.
PROWRITER: Both tractor and friction feed standard.

INTERFACE

MX-80: Parallel interface standard; optional serial interface for about $75 extra.
PROWRITER: Parallel and serial interface standard.

WARRANTY

MX-80: 90 days, from Epson.
PROWRITER: One full year, from Leading Edge.

PRICE

Heh. heh.

Distributed Exclusively by Leading Edge Products, Inc., 225 Turnpike Street, Canton, Massachusetts 02021. Call: toll-free 1-800-343-6833; or in Massachusetts call collect (617) 828-8150. Telex 951-624.

LEADING EDGE.™

For a free poster of "Ace" (Prowriter's pilot) doing his thing, please write us.